Aplicación web de gestión de software "RISS"

Trabajo de fin de grado.

Marina Paz García.

22/06/2015

UNIVERSIDAD CARLOS III DE MADRID.
ESCUELA POLITÉCNICA DE MADRID.
GRADO EN INGENIERÍA TELEMÁTICA.

AUTOR: Marina Paz García.

TUTOR UC3M: Jesús Arias Fisteus.

EMPRESA CLIENTE: Iri Worldwide

TUTOR EMPRESA: Oscar García Noblejas.

Resumen

En la actualidad, la informatización de los servicios en las empresas aumenta día a día, esto generalmente da lugar a la reducción de tiempo y como consecuencia de dinero. La reducción de costes es el objetivo prioritario de cualquier entidad, cualquier ayuda será bienvenida. Si se traduce esto al sector de las telecomunicaciones, supone facilitar el trabajo al usuario final.

Las aplicaciones de gestión, listado o manipulación de productos son las más demandadas por el mundo empresarial. Su finalidad es deshacerse de los cientos de correos entre usuarios o el papeleo de contratos para la gestión de estos servicios. La solución: la informatización de estas funciones.

El Proyecto *RISS* soluciona parte de este problema. Es una aplicación de gestión y petición de *software* por parte de usuarios a los servicios de ***helpdesk*** de cada país. El encargo de la plataforma se ha realizado por parte de consultoría multinacional IRI Worldwide, con sede en España, donde se realizaron las prácticas curriculares, con fecha de inicio el 9 de Junio de 2014 hasta el 28 de febrero de 2015. El Proyecto *RISS* (Request to install *Software* and Service) o ***Aplicación web de gestión de software*** se ha diseñado para el departamento de informática y telecomunicaciones de dicha empresa.

El objetivo de la plataforma es reducir el tiempo y evitar el tráfico masivo de correos entre usuarios, servicios de telecomunicaciones y el departamento financiero para la petición de instalación de licencias, así como la aprobación del gasto y *software* en ordenadores propios o de departamento.

Para generar la plataforma se ha creado una página *web* con diferentes vistas: usuarios, *helpdesk* y financieros. Se han utilizado herramientas como HTML, MySql o PHP, las cuales se comentan con más detalle en el anexo de la memoria: Estado del arte.

La aplicación se puso en funcionamiento en la consultoría a principios del mes de febrero. Donde a través del uso de los propios usuarios se irán depurando y corrigiendo errores de uso detectados, así como innovaciones y mejoras.

En la siguiente ficha se puede ver el resumen de las características básicas del proyecto.

Nombre del proyecto	Proyecto Riss (*Request to install Software and Service*) o Aplicación *web* de gestión de *software*.
Fecha de inicio	1 de noviembre de 2014
Fecha de finalización	1 de febrero de 2015
Horas aproximadas de trabajo	360 horas
Autor	Marina Paz García
Tutor de la UC3M	Jesús Arias Fisteus
Empresa cliente	Iri Worldwide, http://www.iriworldwide.com/
Tutor de la empresa cliente, IRI Worldwide	Oscar García Noblejas

Tabla 1.Ficha técnica del proyecto

Abstract

The computerization of services in companies is transforming and increasing daily, with the objective being a reduction of time and cost. The primary objective of any organization will be to reduce cost as much as possible. If it were applied to the telecommunication industry it would result in a more user-friendly product.

Management *software*, listed or handling of products are the most in demand applications by the business world. The purpose is to reduce the amount of mail and contract paper work used in the management of these services. To solve these issues, a higher level of computerization is needed.

The Request to Install Software and Service (RISS) project solved this problem. This is an application of request to install *software* and services by the final users to the services of helpdesk of their country. This platform has made by the multinational IRI Worldwide Company, headquartered in Spain, where the curricular practices were performed, started at 9 of June, 2014 until 28 February 2015. The RISS Project is designed for computers and telecommunications department of the company.

The aim of the platform is to reduce the time and avoid the high traffic with emails between users, helpdesk and financial department. RISS can be used for the request of installing licenses, approving costs or considerations in buying software for companies departments.

To build the platform a website has been created with different views: users, helpdesk and financial. The tools used to create the web application were: HTML, MySQL or PHP. This is discussed in more detail in the appendix "Estado del arte".

The application will be ready for consultancy at the beginning of February. The debugging and correction of errors will be achieved through the use of the application. Also improvements will be implemented with user feedback.

Índice

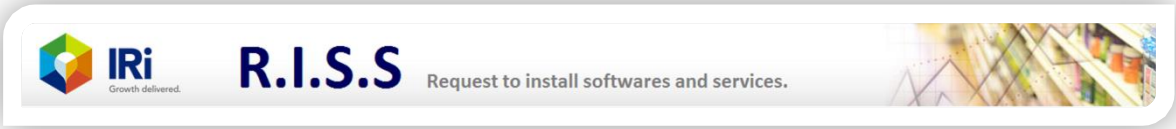
Tabla de Gráficas

Tabla de Ilustraciones y Figuras

Índice de Tablas

Tabla de Anexos:

Capítulo 1: Motivación del proyecto RISS



1.1. Introducción

Vivimos en la sociedad de la información y la informatización de las tareas más básicas comienza a ser algo necesario para el desarrollo adecuado de estas. Se necesita la ayuda de la tecnología y comienzan a olvidarse los métodos convencionales como el envío de información en papel. Se acostumbra a que las esperas sean más cortas cada día. Quedan atrás varios días de espera para una sola respuesta. La inmediatez del traslado de información se traduce en una empresa en reducción de costes y optimización de recursos, lo cual hace más necesaria esta técnica.

Formularios de validación, páginas *web* o aplicaciones para móvil o Tablet son unas de las soluciones a estos problemas. Priorizan la importancia de cada gestión y permiten dar al usuario final una respuesta rápida, evitando a los operarios la necesidad de contacto directo con los clientes.

El proyecto cuyo nombre en clave es "**RISS**", tiene como fin el diseño, implementación y puesta en servicio de una aplicación *web* para la gestión de *software*, solicitada por la empresa cliente, la consultora IRI Worldwide.

Dicha aplicación tendrá una finalidad de soporte y refuerzo, permitiendo al usuario agilizar un proceso antes mucho más costoso, tanto en tiempo como en recursos. Asimismo, las peticiones de usuario y sus correspondientes modificaciones y aprobaciones serán añadidas por usuarios con privilegios especiales llamados "editores", quienes además de este papel tendrán el de valorar, a través de gráficos, el número de peticiones o el presupuesto gastado al final del año por cada país para su reducción o ampliación de crédito.

El proyecto comenzó con la asignación de recursos por parte de la empresa cliente, tras el cual se realizó la planificación del mismo. Una vez definidos los requisitos de la aplicación, el diseño de la misma. Se implementa en bloques individuales para la posterior integración del sistema completo. Tras completar la integración y con el sistema en funcionamiento, éste será testado mediante una batería de pruebas modulares previamente diseñada. Finalmente, se entrega el *software* junto con la documentación pertinente para su posterior evaluación al cliente.

Por parte de la empresa cliente se asume que los integrantes del equipo tienen soltura con el desarrollo de aplicaciones en la programación de páginas *web*, así como con la gestión eficiente del tiempo para la finalización de tareas antes de las fechas límites

establecidas. De igual manera, se conoce que los integrantes del equipo de trabajo llevan una carga de actividades externas al proyecto que consumen gran parte de su tiempo y que pueden ser limitantes para el mismo; no obstante, se espera que esta limitación no sea motivo de retraso en las entregas correspondientes o de empobrecimiento de la calidad del *software*.

1.2. Objetivos

El objetivo de este proyecto es el desarrollo de una aplicación web de gestión de *software* que permita satisfacer plenamente las demandas requeridas por la empresa cliente. La creación del *software* "RISS", de un modo paralelo pretende demostrar los conocimientos adquiridos tanto en el área tecnológica como el trabajo que lo compone en la gestión y elaboración de un proyecto de *software* de tamaño grande, para así estar preparado a la incorporación al mundo profesional del *software* una vez acabado dicho periodo de instrucción.

Las funciones básicas de la aplicación web se pueden resumir:

- Control y mantenimiento de los proyectos de gestión de *software* y licencias de la empresa.
- Asignación de personal, registro de costos y tiempo dedicados a dicho proyecto.
- Gestión de costes imputados a los proyectos.
- Análisis por fechas y país de gastos realizados.
- Posibilidad de exportación de datos a Excel en modo de factura.

La plataforma debe ser segura, cumplir con los requisitos de privacidad y seguridad de los usuarios finales, sin revelar información y no haber brechas en la protección de datos o contraseñas.

En definitiva, el objetivo principal de la aplicación es la reducción de tiempo del personal de la empresa implicado.

De un modo paralelo, los objetivos del proyecto como trabajo de fin de grado se resumen en:

- Conocer qué es y cómo funciona un entorno *web*
- Aprendizaje de técnicas de implementación con lenguaje PHP, HTML, CSS, JavaScript y MySQL

- Gestión, modificación y tratamiento de datos.
- Diseño y estructuración de un proyecto de tamaño grande para un cliente, la empresa consultora IRI Worldwide.

1.3. Introducción al resto de la memoria

La memoria de la plataforma RISS está dividida en varios capítulos, los cuales se resumen a continuación. Además contendrá una serie de apartados que ayudan a la descripción de dicho capítulo.

Capítulo 1: Introducción

Capítulo 2: Antecedentes

En este capítulo se muestra un breve estudio sobre las tecnologías utilizadas. Los lenguajes de desarrollo el porqué de esos y no otros. El análisis del estado del arte en este ámbito tecnológico.

Capítulo 3: Planificación

Se describe una planificación en tiempo y costes del proyecto. El camino crítico que ha existido y la asignación de recursos por parte del jefe de proyecto.

Capítulo 4: Análisis de la plataforma

Este capítulo contiene el análisis de los requisitos del sistema, tanto funcionales como no funcionales, la arquitectura de la plataforma. El modelo que se ha seguido y porqué. El diseño de la plataforma y aspectos más relevantes de la implementación del mismo.

Por último se incluye un apartado donde se explica la viabilidad del proyecto con una pila de pruebas y validaciones.

Capítulo 5: Conclusiones

Análisis de un marco regulador y un entorno socioeconómico donde la plataforma se desarrolla. Incluye trabajos futuros y conclusiones obtenidas del proyecto.

Capítulo 6: Anexos

El capítulo de anexos se adjuntan: resumen global en inglés, manual de usuario y tablas de precios de forma explicativa.

1.4. Introduction

We live in the information and computerization society and the most basic tasks is becoming a necessity for the proper development of these. We need the help of the technology and the conventional methods of sending information, so sharing information in paper have become obsolete.

It is customary that the waits are shorter every day. We have left behind, the age when we needed several days of waiting for a single answer. The immediacy transfer of information translates into a cost reduction and resource optimization for companies, which makes this technique more necessary.

Forms validation, web pages or applications for mobile or Tablet are one of the solutions for these problems. Prioritizing the importance of each management and allowing the end user to give a quick response, operators avoiding the need for direct contact with customers.

The project whose code name is "RISS", aims to design, implementation and commissioning a web application for managing software, requested by the company client, the consultancy IRI Worldwide.

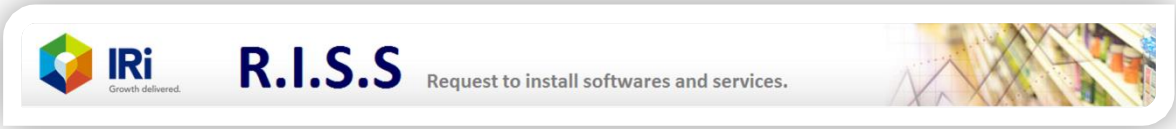
This application will aim to support and reinforce, allowing users to accelerate a previously much more costly process, in time and resources. Otherwise, user request and their alterations or approval will be added by users with special privileges called "editors". These editors can value the request and analyzes the statistics through graphics. The number of requests, the total cost per country or the reduction or amplified of the cost can be evaluated with this method.

The project began with the assignment of resources by the client company, after which the planning it was made. Then, defining the requirements of the application, the design of it, it is implemented in individual blocks for the subsequent integration of the complete system. After completing the integration with the operating system, it will be tested with a battery of modular tests designed previously. Finally, the software supplied with the relevant documentation for further evaluation by the customer.

The part of the client company assumes that team members are familiar with application development programming web pages, as well as the efficient management of time for tasks completion before the deadlines set. Similarly, it is known that members of the team carry a load outside the project would consume much of your

time and can be limiting for the same activities; however, it is expected that this limitation is not the cause for delay in corresponding deliveries or improvement in software quality.

Capítulo 2: Estado del arte



2.1. Antecedentes

Para comprobar la viabilidad de la plataforma, en primer lugar se han comprobado los antecedentes y el estado del arte de las tecnologías a utilizar y la existencia de otros *software* libres o de pago.

En este capítulo se detallan todos los elementos y tecnologías usadas para el desarrollo de la plataforma de aplicación *web,* RISS. Se desarrolla en apartados, donde se explican cada una de las capas en detalle.

En concreto se desarrollan tecnologías relacionadas con:

- Desarrollo de la aplicación *web*
- Envío de peticiones desde el cliente al servidor
- Servidor de base de datos
- Gestión de base de datos.

2.2. Estudio de la viabilidad

Este proyecto se centra en el ámbito de una consultoría, IRI Worldwide. IRI es una empresa consultora de negocios que dedica sus estudios a los sectores de la alimentación y la droguería. Destacan clientes mundialmente conocidos como Coca-Cola, Mercadona o Shweppes entre otros.

En los últimos años se ha producido un giro en la economía de 180º. Ante esta nueva realidad donde los consumidores cambian su comportamiento de compra, los fabricantes y distribuidores no pueden esperar semanas o meses para reaccionar. Para dar una respuesta rápida, surgen consultoras de mercado, como IRI Worldwide, que permite que sus clientes vayan un paso por delante de sus competidores.

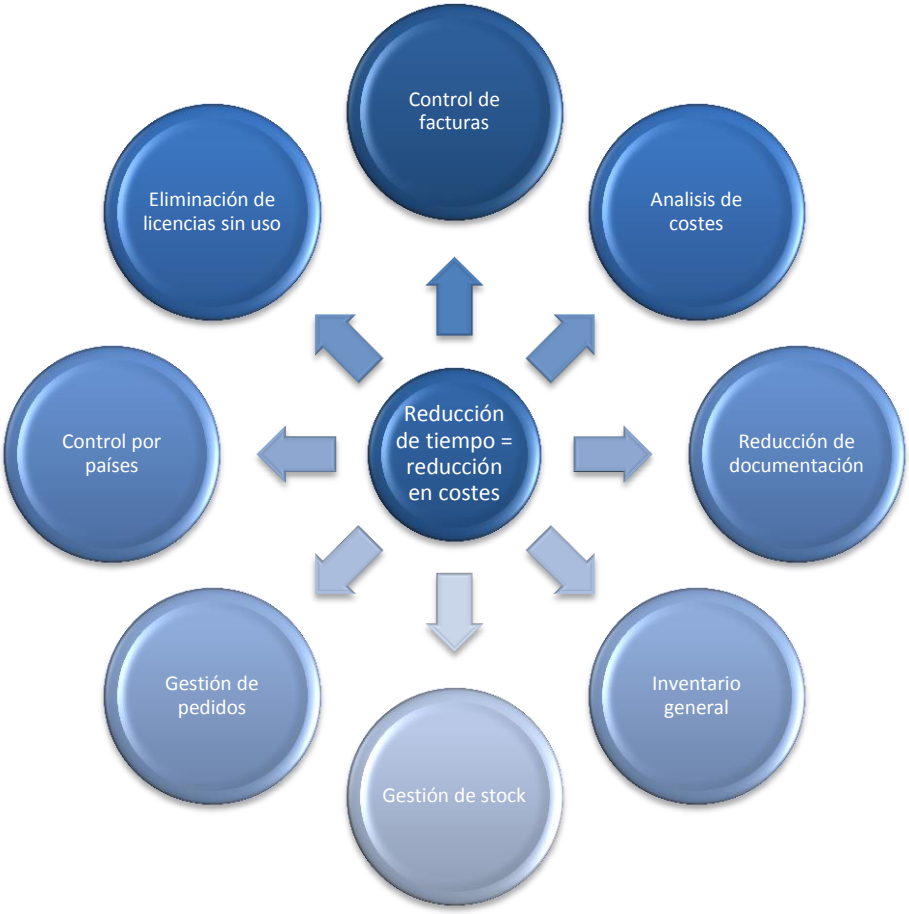
Esta es una empresa americana. Tiene dos sedes en España, las cuales trabajan a la par en proyectos comunes. Se sitúan en Madrid y Barcelona. Su comunicación es fundamental para el buen desarrollo de estudios y trabajos. Los programas de comunicación son clave para su día a día. Esta empresa lleva a cabo proyectos de envergadura internacional, junto con Francia, Italia o incluso la India, donde se sitúa la sede del departamento técnico y de soporte.

El funcionamiento de la empresa se puede resumir en unas pocas líneas. Las empresas esperan recibir la información del punto de venta, para después analizarla.

Una vez examinada, a las pocas semanas los resultados dejarán de ser válidos. El papel principal de IRI es facilitar esta información rápidamente antes de que pierda valor.

En definitiva, la clave del éxito de una consultora es el tiempo de reacción, que a corto plazo se transforma en costes. El papel de un técnico en una empresa como esta es fundamental, pues será él quien pueda transformar la comunicación en un proceso inmediato.

Este proyecto se centra en la comunicación entre varios departamentos para el control de *software*, licencias y por tanto costes. Si bien antes de tomar la decisión de crear una nueva plataforma, se estudia los productos existentes ya en el mercado.



Gráfica 1. Reducción de costes = reducción en tiempo

2.3. Plataformas alternativas

Antes de plantear la posibilidad de crear una nueva plataforma, se han estudiado otros programas de objetivos similares ya existentes en el mercado, de los cuales destacan:

RT (Request Tracker)[1], es probablemente el sistema de gestión de Tickets con más penetración en el mercado. Con licencia GNU y desarrollado en Perl. Es gratuito y con la única necesidad de un servidor Apache y una base de datos. Su problema es que funciona a través de correos electrónicos A pesar de contar con una interfaz gráfica, esta solo será manejada por los administradores, por lo que la solución queda descartada, ya que el envío de correos debe ser meramente informativo, siendo este un requisito indispensable de la empresa.

OTRS, Open ticket request System[2], se ofrece bajo licencia GNU, ofrece facilidades como calendarios, *workflow*, gestión de problemas… El inconveniente está en que desde 2003 está gestionada por una empresa, por lo que será necesario contratar sus servicios de soporte.

OSTICKET[3], GLPI[4], SPICEWORKS HELLP DESK[5], REDMINE[6]…

Todos ellos son *software* de calidad pero el problema en común es que no se ajustan a los requerimientos básicos de la empresa cliente. En primer lugar, todas estas plataformas son *software* para la gestión de incidencias, no para la gestión de *software,* y por ello el flujo de la aplicación no se adapta al deseado. No existe la división por departamentos, no se encuentra la posibilidad de extracción de datos en modo factura, el control de gastos mediante estadísticas, o no existe una reducción real del tiempo con el uso de estas aplicaciones, bien sea por la cantidad de correos electrónicos que envían o por la dificultad que muestran al usuario final. Por todos esos motivos, el proyecto RISS necesita una nueva plataforma diseñada y pensada específicamente para esta empresa.

2.4. Estudio y comparación de lenguajes de programación *web*

El proyecto RISS lleva a cabo la elaboración de una nueva plataforma. Para ello se estudian y comparan varios lenguajes de programación.

Las aplicaciones *web* se desarrollan comúnmente con una arquitectura cliente/servidor. Cada lenguaje será procesado por una de las partes de esta arquitectura. Existen varios lenguajes posibles para la creación de una plataforma *web*. Este estudio se centra en los lenguajes más relevantes.

2.4.1. Lenguajes en el Cliente

En la actualidad el aumento de dispositivos móviles o *tablets* cambia el concepto de la programación de páginas *web*. Las nuevas plataformas deben ser adaptables a cualquier dispositivo, no solo a las pantallas convencionales. Así surge el concepto de diseño *web* adaptable o adaptativo, conocido como *responsive web desing* (RWD). Esta nueva filosofía de diseño tiene como objetivo adaptar las páginas *web* a cualquier dispositivo sin alterar su apariencia. Este concepto nace de la mano de Ethan Marcotte, definiéndolo en el artículo: "Responsive Web Design". A List Appart, issue 306. May 2010

"Designers have coveted print for its precision layouts, lamenting the varying user contexts on the web that compromise their designs. Ethan Marcotte advocates we shift our design thinking to appropriate these constraints: using fluid grids, flexible images, and media queries, he shows us how to embrace the "ebb and flow of things" with responsive web design." [2]

El concepto de *responsive web* pretende que con un único diseño *web* se tenga una visualización en cualquier dispositivo. RWD maneja el flujo de una página *web* mediante el cambio de ventanas. El lenguaje que mejor se adapta a esta nueva filosofía es HTML5.

2.4.1.1. HTML5 (HYPERTEXT MARKEUP LANGUAGE)

HTML es un lenguaje basado en etiquetas, las cuales son interpretadas por los navegadores. Es el principal lenguaje de presentación de contenidos en la *web*.

Existían otros lenguajes de plataformas *web*, destacando Java y Flash, pero su falta de integración y haber sido concebidos desde el inicio como *plug-ins* hacen de ellos una opción a desechar. Por este motivo el lenguaje HTML se es el principal lenguaje de la *web.*

La integración de los tres componentes principales de una interfaz HTML, CSS y JavaScript, esta última permitiendo numerosas innovaciones sobre todo en temas de estética. Se necesita una técnica que unifique todas estas técnicas, surge así el lenguaje HTML5.

"*HTML5 no es una nueva versión del antiguo lenguaje de etiquetas, ni siquiera una mejora de esta ya antigua tecnología, sino un nuevo concepto para la construcción de sitios web y aplicaciones en una era que combina dispositivos móviles, computación en la nube y trabajos en red*." [7]

Ilustración 1. HTML5 Y CSS

Así HTML5 se define como un lenguaje de programación en el cliente, que integra CSS y JavaScript. Por tanto lo convierte en el lenguaje más indicado para el procesamiento de aplicaciones *web responsive. [3]*

Esta nueva tecnología propone estándares para cada aspecto de la *web*. A partir de ahora, HTML ofrece los elementos de estructura, CSS ofrece una estructura única y atractiva a la vista y JavaScript provee dinamismo y estética a la *web*.

2.4.1.2. CSS (CASCADE STYLE SHEETS)

Las páginas *web* precisan estilo y diseño, no solo funcionalidad. El lenguaje CSS trabaja junto con HTML y es incluido en las aplicaciones para definir estilos visuales en el documento, como fondos, imágenes, logos, ventanas…

Un navegador *web* procesa cada elemento HTML como una "caja", entendiendo como una página *web* un conjunto de cajas. A cada se caja pueden aplicar un conjunto de reglas definidas en las llamadas hojas de estilo de CSS.

Los navegadores imponen ciertas reglas sobre cualquier página, la inserción de una hoja de estilo, sobrescribe cualquier regla establecida anteriormente. Estas nuevas reglas forman un modelo de caja, llamado sistema de disposición.

Las mejoras sobre el lenguaje CSS se engloban en la versión CSS3. Su principal intención es la de reducir al máximo nivel el uso de JavaScript. Así, en esta nueva versión cubre además de estilos, como fondos o ventanas, técnicas de diseño de forma y movimiento ya representadas en pantalla [7]. Un ejemplo de ello utilizada en este proyecto es el uso de CSS3 en la cabecera de las páginas. Al pasar el ratón por encima se iluminan ciertas casillas.

2.4.1.3. JavaScript

JavaScript es un lenguaje de programación interpretado, es decir, no es necesaria su compilación para ejecutarlo. En otras palabras, JavaScript puede ejecutarse en cualquier navegador sin necesidad de intermediarios.

Es un lenguaje que se ejecuta en el propio cliente, a la vez que se van descargando las páginas HTML. JavaScript es una marca registrada por Oracle Corporations[1].

"Javascript es un lenguaje de programación interpretado (un lenguaje de tipo *script*). A pesar de que existen intérpretes no dependientes de ningún navegador, es un lenguaje de *script* que suele encontrarse vinculado a páginas web. JavaScript y Java son dos lenguajes de programación distintos con filosofías muy diferentes. El único punto en común es la sintaxis, ya que cuando Netscape diseñó JavaScript, se inspiró en la sintaxis de Java." [1]

Jquery es una librería de JavaScript que facilita una API fácil de usar que funciona en los principales navegadores y que simplifica la tarea de desarrollo. Es capaz de enviar llamadas asíncronas al servido, entre otras funciones. La característica principal de la biblioteca utilizada es que permite cambiar el contenido de la página mostrada al cliente sin necesidad de recargarla.

2.4.1.4. Highcharts. Los gráficos

Para el mundo de los negocios tener siempre presente las estadísticas es una de las claves para el control de recursos, tiempo o costes.

Una vez introducida el lenguaje JavaScript en el punto anterior, se va a profundizar en este apartado en una de las librerías de JavaScript, se trata de Highcharts JS[1][8]. Esta API permite implementar multitud de opciones para generar estadísticas gráficas, desde la gráfica de barras hasta gráficas 3D.

Para utilizar esta librería únicamente necesitamos incluir la librería JavaScript y definir una nueva instancia de Higthcharts.

Esta librería usa tanto Jquery como MooTools[2] para las tareas JavaScript más frecuentes. Además, Internet Explorer necesita ExCanvas[3] que emula al elemento Canvas.

2.4.2. Lenguajes en el servidor

En la parte del servidor será donde se ejecute la lógica de la aplicación y sus interacciones con la base de datos si es necesaria.

En este campo existen más variedad de lenguajes a elegir y no está tan claro, como en el caso de HTML, cual es la mejor opción. Para evaluar todas las opciones se adjunta una tabla comparativa de lenguajes.

2.4.2.1. Tabla comparativa de lenguajes

En esta tabla se reflejan a modo de resumen los lenguajes más relevantes.

Lenguaje	Ventajas	Desventajas
PHP	- Lenguaje libre - Soporte para gran cantidad de datos - Integración con varias bibliotecas - Procesamiento de información con, manipulación de cookies, páginas dinámicas… - Curva de aprendizaje rápida - Lenguaje multiplataforma - Soporta orientación a objetos - Puede ser combinado con HTML - No se necesita instalación PHP en el lado del cliente	- Necesidad de un servidor web - No ofrece la misma organización en capas que JAVA o .NET - Todo el trabajo se realiza en un servidor y las solicitudes pueden ser ineficientes.
ASP.NET	- Programación declarativa - Rica en FRAMEWORKS - Mayor seguridad - Incremento de la velocidad	- Lenguaje de pago - Tienen que correr en ordenadores con Windows y un servidor Web
RUBY ON RAILS	- Orientado a objetos - Maneja excepciones - Puede cargar librerías - Software libre - Multiplataforma	- No cuenta con gran cantidad de documentación (como por ejemplo PHP)

Python	- Código interpretado - Software libre - Multiplataforma - Orientado a objetos	- Relativamente lento, por ser un lenguaje interpretado

Estos datos son el resumen básico de las características de los principales lenguajes. Tras el estudio de estos se deduce que una la mejor opción para la programación de la lógica de la plataforma es PHP.

"Among the most popular server-side languages are PHP, Perl, Ruby, ASP.NET, Python, Java and ColdFusion. Some are proprietary, such as ASP.NET and ColdFusion, while Perl and PHP have the advantage of being open source, free, and available for servers that run either Linux or Windows. Consequently, they are offered by almost all web hosting companies. Of the two, PHP is easier to learn than Perl, and more widely used. At the time of writing, hotscripts.com, one of the best places to find ready-made web programs, offers 4,000 Perl programs and 15,000 PHP ones for download." [4]

Cabe destacar que aun tras el estudio de varios lenguajes, la empresa cliente especifica dentro del documento de requisitos que la plataforma debe ser generada con PHP Y MySQL en el lado del cliente.

2.4.2.2. PHP (Hypertext Preprocessor)

PHP (*Hypertext Preprocessor*) es un lenguaje de código abierto especialmente creado para el desarrollo web y que puede ser añadido en aplicaciones con HTML.

Este lenguaje permite incrustar códigos HTML. Esta será la principal diferencia ante otros lenguajes como Perl[5] o C[6].

Desde el punto de vista del cliente su principal diferencia con código JavaScript, por ejemplo, es que el código es ejecutado en el servidor, generando un HTML y enviándolo al cliente. Es muy versátil a la hora de programar, permite que todo el código HTML que se muestre al cliente sea el que reciba de ejecutar el script en el servidor, sin que el usuario final sepa qué hay detrás de este.

El código PHP, está encerrado en etiquetas de comienzo y final *"<?php ……..?>"* permitiendo salir o entrar del código a través estas llaves.

Este lenguaje está orientado principalmente a la programación de script del lado del servidor. Por ejemplo, se utiliza para compilar datos de formularios, generar páginas con contenidos dinámicos, enviar y recibir *cookies*, manejar sesiones, etc. Limitarse a generar páginas HTML no es su única función. Destacan entre otras la creación de imágenes, ficheros PDF e incluso películas Flash (usando libswf y Ming).

De todas las funciones, la plataforma RISS toma como puntos clave de su diseño la creación de ficheros Excel, con el API de PHPExcel[6] junto con el soporte de protocolos como HTTP para el envío y recepción de peticiones y soporte de protocolo SMTP para el envío de correos.

2.4.3. Capa de datos

Un gestor de bases de datos es un sistema que permite la creación de tablas, donde almacenar información y poder administrarla. El propósito de cualquier sistema de gestión de datos es manejar de forma clara e intuitiva cierta información, de entre sus plataformas destacan:

De pago:

- Oracle
- dBase
- MySQL (edición de pago)

Gratuitas:

- SQL Server
- MySQL

La gestión de la información por parte del cliente permite la abstracción, facilitando al usuario final ahorrarle detalles de almacenamiento, así como independencia. Se deberá tener en cuenta el control de la concurrencia, ya que típicamente será más de un único usuario quien interaccione con la información, por lo que es necesario el control concurrente del acceso a los datos. La seguridad de los datos, este último punto destacable en cualquier empresa. Aun así esta gestión conllevará la desventaja de necesitar *hardware* adicional y un administrador para la base de datos.

"In the world of databases, the most popular language for interrogating a database is something called Structured Query Language, or SQL. Many database products use

SQL, or something like it. Most SQL or SQL-like database server products cost serious money, such as Oracle or Microsoft SQL Server. One, though, is totally free of charge, and unsurprisingly this is the one that most web hosting companies install for your use. The product is called MySQL (pronounced my ess queue ell, or sometimes my sequel). Don't be put off by the price tag – MySQL is corporate-strength software, just as happy storing tens of millions of items as a mere handful." [4]

Tras el estudio de varios gestores, en este proyecto se utilizó MySQL gratuito, siendo este uno de los requisitos de la empresa cliente.

2.4.3.1. MySQL

MySQL es la base de datos de código abierto de mayor aceptación mundial. Permite la creación de bases de datos para web con alto nivel de fiabilidad.

Esta plataforma de código abierto se basa en una arquitectura cliente/servidor, donde el cliente inicia la petición y esta es recogida por el servidor, quien la gestiona conectado con la base de datos. Es en este punto donde entra en juego MySQL, que permite:

- Definir la base de datos a partir de varias tablas
- Estas tablas pueden estar relacionadas entre sí siguiendo un modelo de entidad-relación
- Almacenar datos en estas tablas
- Codificar o encriptar ciertos datos relevantes, como contraseñas
- Crear una lógica de negocio que permita combinar y calcular dichas tablas.

Una aplicación *web* suele necesitar la comunicación con un servidor de bases de datos, donde se aloja la información de la aplicación. Para ello un programa que utilice PHP, necesitará realizar un conjunto de tareas que le permitan conectarse a esas bases de datos. Será necesaria la instalación de una API que permita y maneje esas conexiones. A este *software* se le conoce como conector, que en PHP se proporciona como una extensión.

La extensión de MySQL está obsoleta desde la utilización de PHP 5.5.0, por lo que en esta plataforma y con vistas al futuro se utilizó la extensión MySQLi. Dentro de las características de MySQLi o MySQL mejorado nos encontramos que es una interfaz orientada a objetos.

2.4.4. Estructura final, cliente/servidor y petición/respuesta

Si se unifican todas las plataformas vistas hasta ahora, tanto del cliente como del servidor, podemos resumir la estructura de una aplicación *web* en dos partes claramente diferenciadas, cliente y servidor.

Se han estudiado las técnicas del lado del cliente, que en resumen son las propias de la interfaz web, quien genera la petición y las técnicas del cliente, quien sirve y crea una respuesta. En la siguiente figura se puede observar un esquema de esta organización:

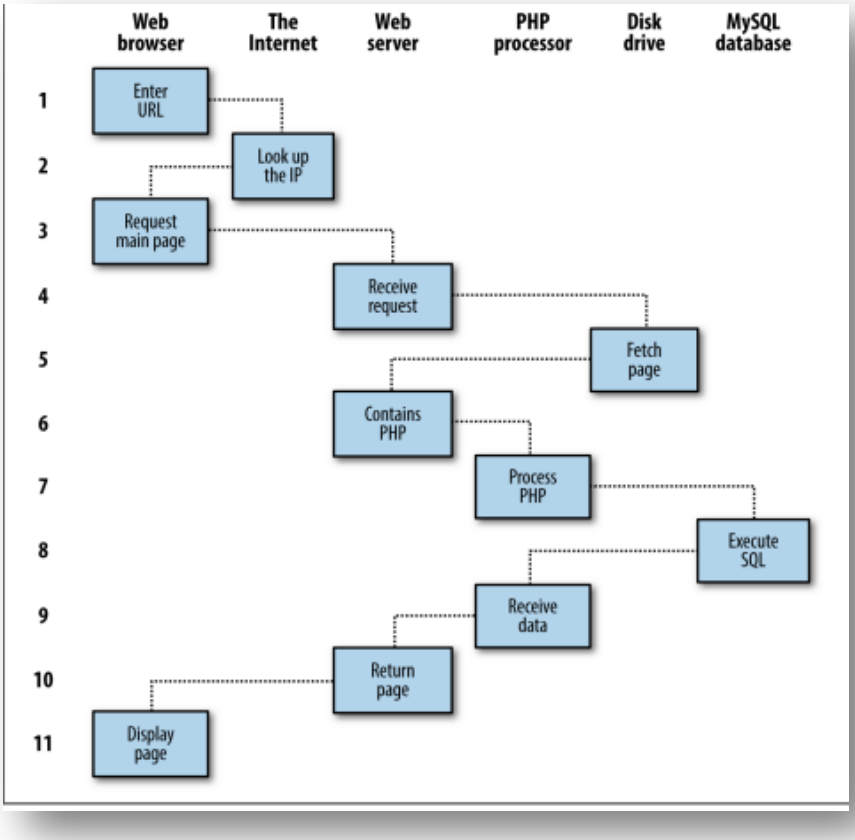


Ilustración 2.Figura cliente/servidor y petición/respuesta. Obtenida del Libro "Learning PHP, JavaScript, CSS y HTML5", O' realLy

Dentro de este entramado, queda resolver la pregunta de cómo se comunican entre sí. Es aquí donde entra en juego el estudio de protocolos de internet.

El servidor está a la escucha de una petición y a la espera de enviar una respuesta través de un puerto de TCP. El protocolo seguido para este intercambio es el protocolo

HTTP (*HyperText Transfer Protocol*), según lo especificado en el RFC 2616, por el cual se realiza el intercambio de información.

HTTP es un protocolo de solicitud-respuesta que opera sobre TCP. Especifica qué mensajes pueden enviar los clientes y recibir los servidores. Los encabezados de la petición se proporcionan en ASCII, al igual que SMTP.

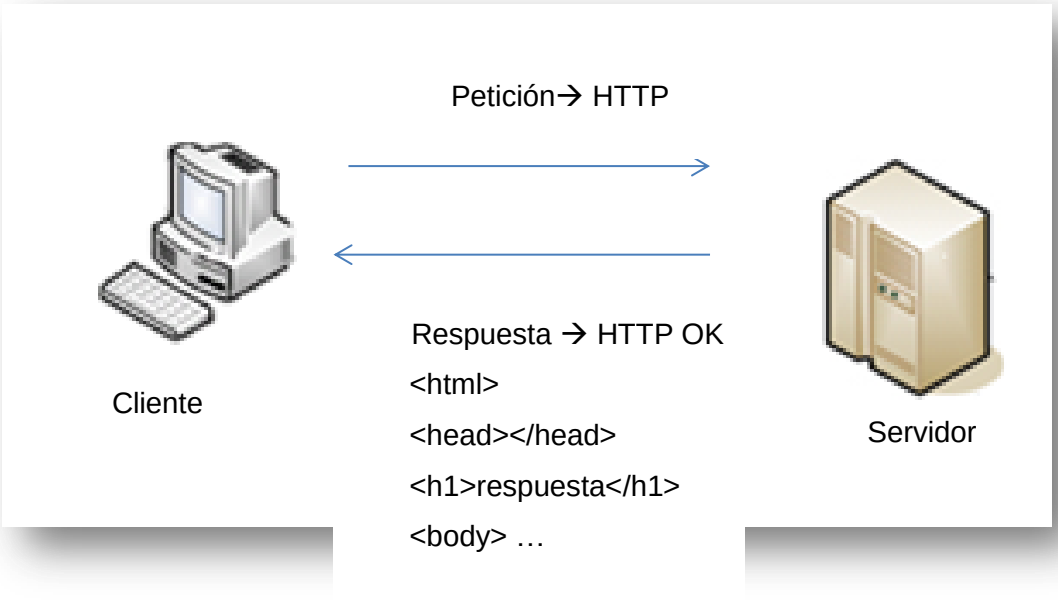


Ilustración 3. Protocolo http

Normalmente las peticiones por parte del cliente se realizan mediante la conexión TCP por el puerto 80 en la máquina del servidor. El protocolo TCP es el encargado de manejar los mensajes, la confiabilidad, el control de congestión o la llegada ordenada de datos.

Los recursos se obtienen mediante una conexión persistente. Esto es, la conexión TCP se abre al inicio, después se envían las solicitudes que se deseen (por ejemplo, tres imágenes en una misma página) y se cierra.

Una aplicación *web* es un conjunto de llamadas o rutinas, cada una a través de una URL (*Uniform Resource Locator*). Esta URL tiene tres partes diferenciadas: el protocolo, el nombre DNS de la máquina que se encuentra y la ruta donde se encuentra la página. Esta ruta tiene un nombre jerárquico que modela la estructura del directorio de archivos en el servidor.

Cuando en el lado del cliente el usuario final hace accede a un hipervínculo, el navegador lleva a cabo una serie de pasos para enviar la petición y recibir la respuesta.

1. Identificar la URL.
2. El navegador pide al servidor de DNS la dirección IP del servidor.
3. El servidor de DNS responde.
4. El navegador realiza una conexión TCP a esa dirección IP por el puerto 80, puerto que se utiliza para el protocolo HTTP por defecto.
5. El cliente envía una solicitud HTTP para solicitar el recurso indicado en la ruta. Los recursos pueden ser páginas HTML, hojas de estilo CSS, programas JavaScript, imágenes, etc.
6. El servidor envía el recurso solicitado en la ruta como respuesta HTTP.
7. El cliente recibe la respuesta y la interpreta.
8. Se libera la conexión TCP si existe inactividad en ese servidor durante un periodo de tiempo. [5]

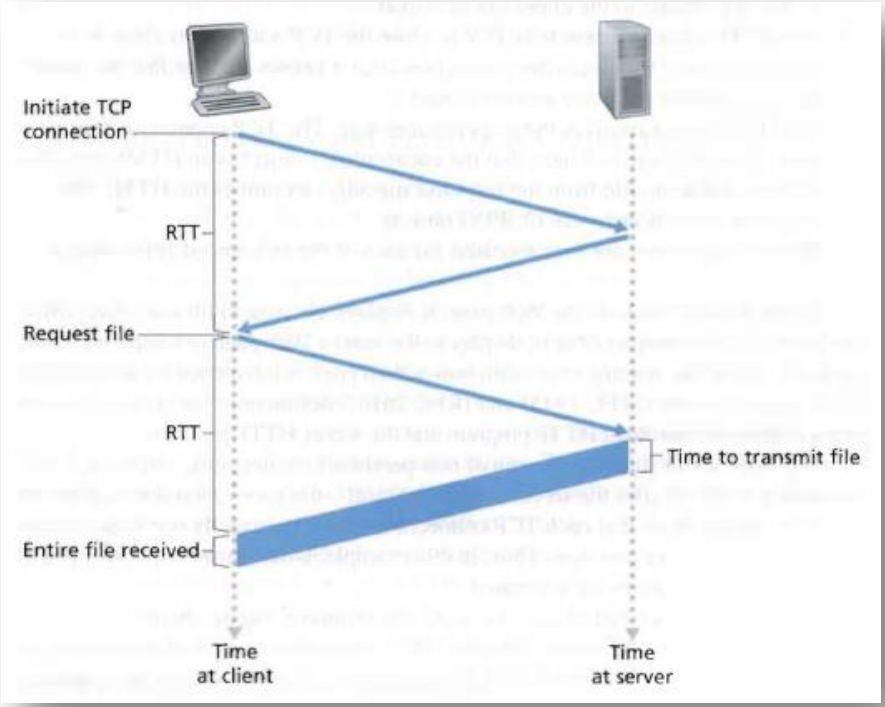
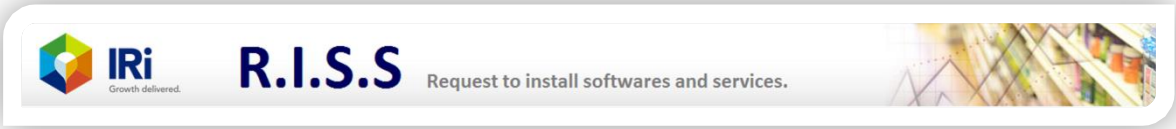


Ilustración 4. HTTP y TCP. Imagen obtenida de Computing Networking, Kurose

Capítulo 3: Planificación



3.1. Plan de trabajo

3.1.1. Metodología de desarrollo

Se ha utilizado un modelo en cascada. Este modelo trata del desarrollo del *software* por etapas, distintas y separadas. Se necesita la especificación previa de los requisitos por parte de la empresa cliente. Cada fase del proyecto debe esperar a la finalización del anterior. [9]

El principal problema del modelo en cascada es que hace difícil responder al cambio de requerimientos del cliente ante la inflexibilidad de dividir de nuevo el proyecto en etapas. Se entiende que muy pocos sistemas de *software* utilizan el modelo en cascada, pero lo hace apropiado para este proyecto, ya que los requisitos se cierran y están bastante limitados en el proceso de diseño por parte de la empresa.

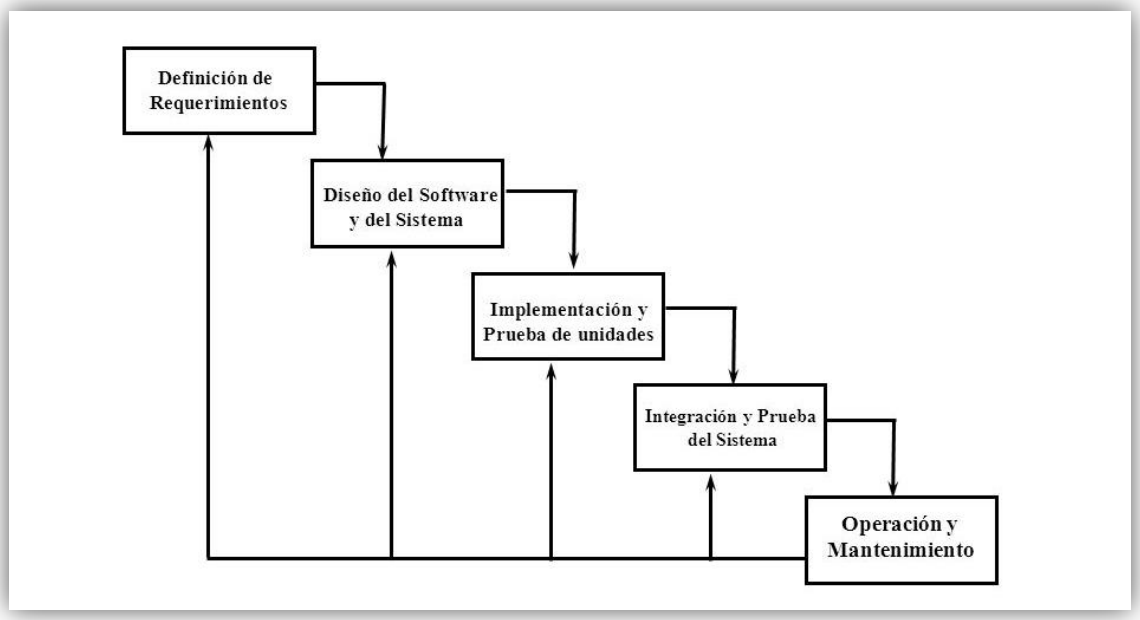


Figura 5. Modelo en cascada, Ian Somerville 2002.

En la figura se representa el esquema básico que sigue el modelo en cascada. Donde una fase tiene que estar completa antes de pasar a la siguiente. Comienza por la definición y análisis de requisitos, el diseño del *software* y del sistema, implantación y pruebas. Integración de las pruebas del sistema y operación y mantenimiento del proyecto.

3.1.1.1. Análisis de requisitos

La empresa cliente es la encargada de detallar minuciosamente todos los requisitos que debe tener su sistema. Tras la evaluación y el descarte de los algunos requisitos. Se procede a su definición.

Resumiremos los requisitos con la división en tres usuarios y por tanto tres vistas diferentes de la aplicación. Usuario final, usuario *helpdesk* y usuario del departamento financiero.

En el apartado de requisitos se detallaran con más precisión cada uno de ellos.

3.1.1.2. Diseño del software y del sistema

En esta etapa se organiza el sistema y los elementos que lo componen, definiendo la estructura de la solución a implementar y la arquitectura elegida.

A través del patrón elegido, modelo vista controlador o MVC se establece un diseño que cumple estas características.

3.1.1.3. Implementación y pruebas de unidades

En esta fase se lleva a cabo la implementación de la solución dada en la etapa anterior.

3.1.1.4. Integración y prueba del sistema

En esta etapa, los elementos programados se ensamblan para componer el sistema y comprobar mediante una serie de pruebas que cumple los requisitos definidos en etapas anteriores.

Se construye en el servidor dado por el cliente y se comenzarán con las pruebas de verificación para su futura puesta en marcha.

3.1.1.5. Mantenimiento

En esta etapa se realiza un mantenimiento del sistema debido a que el usuario final puede no hacer un uso ideal del sistema proporcionado, o bien porque pueden surgir errores no detectados previamente.

3.1.2. Planificación

Para definir los paquetes de trabajo se sigue un criterio basado en la división por actividades. En primer lugar los seis paquetes de trabajo principales son:

- Administración del proyecto (WP_1)

- Análisis de requisitos (WP_2)
- Diseño del proyecto (WP_3)
- Implementación del código (WP_4)
- Pruebas (WP_5)
- Redacción de la memoria (WP_6).

Cada paquete de trabajo tiene una duración adecuada y una fecha límite de finalización marcada con diferentes hitos. La fecha límite de todos los paquetes de trabajo se determinó por la empresa cliente, exceptuando la entrega de la memoria.

Todos los paquetes de trabajo se subdividen en tareas, las cuales marcan el paquete completo. El paquete de trabajo WP_1 se centra en el estudio del problema y de la viabilidad de su solución. En este caso, se produce el estudio del problema, descrito en este documento. Tras la acumulación de correos electrónicos y el problema de no tener una gestión adecuada de la petición de *software* por parte de los usuarios, se decide implementar una plataforma. Tras su estudio se comienza con la documentación y captura de requisitos por parte de la empresa cliente. En este caso serán necesarias varias reuniones con el jefe del proyecto para el acuerdo y la viabilidad de esos requisitos. La asignación de recursos, como el puesto de trabajo con un ordenador, hasta los elementos necesarios para la elaboración del proyecto como licencias o el servidor especificado. Tras la elaboración de un plan, se procede a su planificación en tiempo, que es un punto clave ya que esto se asocia a dinero y a recursos.

Aunque ya se plantean los requisitos básicos en el paquete WP_1, se asigna una parte del tiempo al análisis de los requisitos impuestos por el cliente. En el WP_2 se describen todos los requisitos tanto funcionales como no funcionales, del sistema y de interfaz. También se describe las funcionalidades de usuario.

Llegados a este punto comienza a la elaboración y planteamiento del proyecto. En el paquete de trabajo WP_3 se diseña el proyecto, tanto su estructura externa como interna, desde la base de datos hasta el diseño de los algoritmos. Una vez acabado el diseño solo queda implementar el código y la redacción de la memoria. Este último trabajo no pertenece al ámbito de la empresa.

El proyecto, como se ha mencionado, se realizó con un único ingeniero que hizo el papel de analista, director de proyecto, desarrollador y programador. Por tanto, la asignación de roles será única. Los integrantes del proyecto fueron la empresa cliente y el ingeniero. De un modo paralelo, nos encontramos en la asignación de recursos al

Ingeniero Tutor de la UC3M, Ingeniero Tutor de la empresa IRIWorldWide y el Director de recursos humanos de la empresa cliente. Las funciones de estos últimos integrantes también se detallan en la gráfica.

En la siguiente tabla se presenta la división de los trabajos por paquetes y subtareas, así como su duración y los recursos asignados.

Tabla 2. División de paquetes de trabajo

Nombre de tarea	Trabajo	Duración
APLICACIÓN *WEB* DE GESTIÓN DE *SOFTWARE* RISS	560,4 horas	46,88 días
WP_1: Administración del proyecto RISS	65,4 horas	7,88 días
T1 .1 Estudio del problema	24 horas	24 horas
Director del Proyecto	24 horas	
T1 .2 Documentación	10,4 horas	8 horas
Director del Proyecto	8 horas	
Director Recursos Humanos Empresa cliente	2,4 horas	
T1 .3 Captura de requisitos	8 horas	8 horas
Director del Proyecto	4 horas	
Ingeniero Senior Tutor IRI WORLD WIDE	4 horas	
T1 .4 Asignación de recursos	5 horas	5 horas
Director del Proyecto	5 horas	
T1 .5 Alcance del proyecto	5 horas	5 horas
Director del Proyecto	5 horas	
T1 .6 Planificación	8 horas	8 horas
Director del Proyecto	8 horas	
T1 .7 Riesgos	5 horas	5 horas
Director del Proyecto	5 horas	
WP_2. Análisis de requisitos	59 horas	7,38 días
T2.1 Funcionalidades de usuario y sistema	8 horas	8 horas
Analista	8 horas	
T2.2 Casos de uso	16 horas	16 horas
Analista	16 horas	
T2.3 Requisitos de sistema, usuarios y dominio	24 horas	24 horas
Analista	24 horas	

T2.4 Requisitos de producto organización y externos	3 horas	3 horas
Analista	3 horas	
T2.5 Requisitos de la interfaz	8 horas	8 horas
Analista	8 horas	
WP_3 Diseño del proyecto	64 horas	8 días
T3.1 Diagrama jerarquías UML	8 horas	8 horas
Diseñador	8 horas	
T3.2 Diseño BBDD	24 horas	24 horas
Diseñador	24 horas	
T3.3 Diseño de algoritmos	32 horas	32 horas
Diseñador	32 horas	
WP_4 Implementación del código	138 horas	17,25 días
T4.1 Implementación modelo	42 horas	42 horas
Programador	42 horas	
T4.2 Implementación vista	42 horas	42 horas
Programador	42 horas	
T4.3 Implementación controlador	54 horas	54 horas
Programador	54 horas	
WP_5 Pruebas	64 horas	8 días
T5.1 Diseño de plan de pruebas	8 horas	8 horas
Analista	8 horas	
T5.2 Depuración	24 horas	24 horas
Analista	24 horas	
T5.3 Pruebas de módulos individuales	8 horas	8 horas
Analista	8 horas	
T5.4 Pruebas del sistema completo	24 horas	24 horas
Analista	24 horas	
WP_6 Redacción de la memoria	170 horas	20,63 días
T6 .1 Documentación requerida	120 horas	60 horas
Director del Proyecto	60 horas	
Ingeniero Senior Tutor UC3M	60 horas	
T6 .1 Manual de usuario	30 horas	15 horas

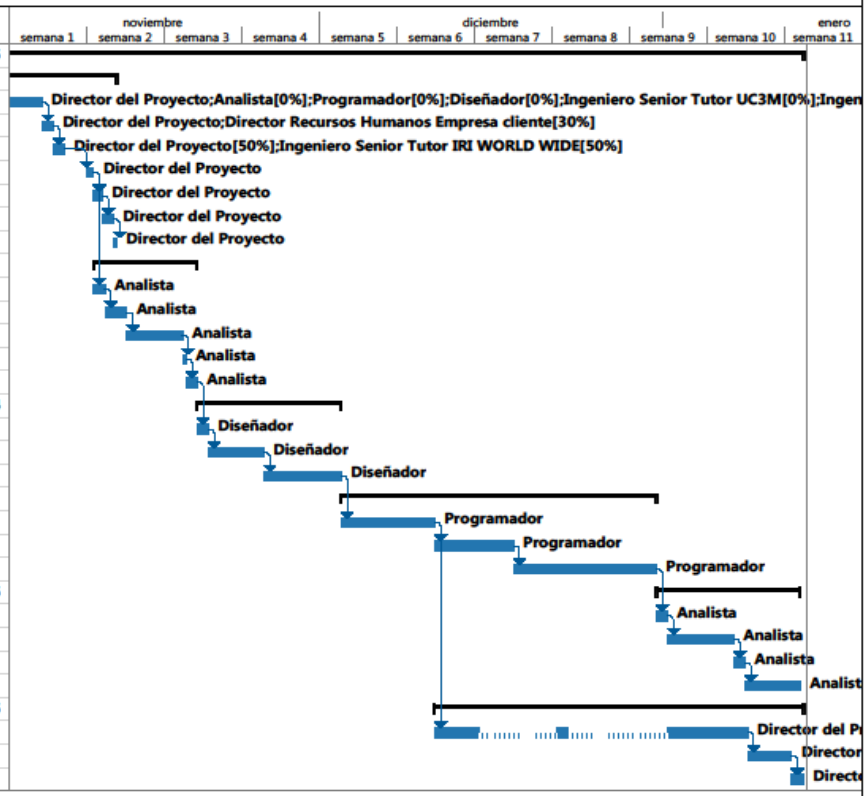
Director del Proyecto	15 horas	
Ingeniero Senior Tutor UC3M	15 horas	
T6 .1 Manual de prototipo	20 horas	10 horas
Director del Proyecto	10 horas	
Ingeniero Senior Tutor UC3M	10 horas	

En la figura adjunta de abajo, se puede observar de forma gráfica el diagrama de Gantt. La asignación de los recursos y el tiempo empleado en cada tarea.

Este diagrama ha sido creado con la herramienta Microsoft Project Professional (2013).[15]

Diagrama de Gantt, Aplicación web de gestión de software -RISS-

Id	Modo de tarea	Nombre de tarea	Duración	Comienzo	Fin
1		**APLICACIÓN WEB DE GESTIÓN DE SOFTWARE RISS**	**46,88 días**	**lun 03/11/14**	**mar 13/01/15**
2		**WP_1: Administración del proyecto RISS**	**7,88 días**	**lun 03/11/14**	**mié 12/11/14**
3		T1 .1 Estudio del problema	24 horas	lun 03/11/14	mié 05/11/14
4		T1 .2 Documentación	8 horas	jue 06/11/14	jue 06/11/14
5		T1 .3 Captura de requisitos	8 horas	vie 07/11/14	vie 07/11/14
6		T1 .4 Asignación de recursos	5 horas	lun 10/11/14	lun 10/11/14
7		T1 .5 Alcance del proyecto	5 horas	lun 10/11/14	mar 11/11/14
8		T1 .6 Planificación	8 horas	mar 11/11/14	mié 12/11/14
9		T1 .7 Riesgos	5 horas	mié 12/11/14	mié 12/11/14
10		**WP_2. Análisis de requisitos**	**7,38 días**	**lun 10/11/14**	**mié 19/11/14**
11		T2.1 Funcionalidades de usuario y sistema	8 horas	lun 10/11/14	mar 11/11/14
12		T2.2 Casos de uso	16 horas	mar 11/11/14	jue 13/11/14
13		T2.3 Requisitos de sistema, usuarios y dominio	24 horas	jue 13/11/14	mar 18/11/14
14		T2.4 Requisitos de producto organización y externos	3 horas	mar 18/11/14	mar 18/11/14
15		T2.5 Requisitos de la interfaz	8 horas	mié 19/11/14	mié 19/11/14
16		**WP_3 Diseño del proyecto**	**8 días**	**jue 20/11/14**	**mar 02/12/14**
17		T3.1 Diagrama jerarquías UML	8 horas	jue 20/11/14	jue 20/11/14
18		T3.2 Diseño BBDD	24 horas	vie 21/11/14	mar 25/11/14
19		T3.3 Diseño de algoritmos	32 horas	mié 26/11/14	mar 02/12/14
20		**WP_4 Implementación del código**	**17,25 días**	**mié 03/12/14**	**mié 31/12/14**
21		T4.1 Implementación modelo	42 horas	mié 03/12/14	jue 11/12/14
22		T4.2 Implementación vista	42 horas	jue 11/12/14	jue 18/12/14
23		T4.3 Implementación controlador	54 horas	jue 18/12/14	mié 31/12/14
24		**WP_5 Pruebas**	**8 días**	**mié 31/12/14**	**mar 13/01/15**
25		T5.1 Diseño de plan de pruebas	8 horas	mié 31/12/14	jue 01/01/15
26		T5.2 Depuración	24 horas	jue 01/01/15	mié 07/01/15
27		T5.3 Pruebas de módulos individuales	8 horas	mié 07/01/15	jue 08/01/15
28		T5.4 Pruebas del sistema completo	24 horas	jue 08/01/15	mar 13/01/15
29		**WP_6 Redacción de la memoria**	**20,63 días**	**jue 11/12/14**	**mar 13/01/15**
30		T6 .1 Documentación requerida	60 horas	jue 11/12/14	jue 08/01/15
31		T6 .1 Manual de usuario	15 horas	jue 08/01/15	lun 12/01/15
32		T6 .1 Manual de prototipo	10 horas	lun 12/01/15	mar 13/01/15

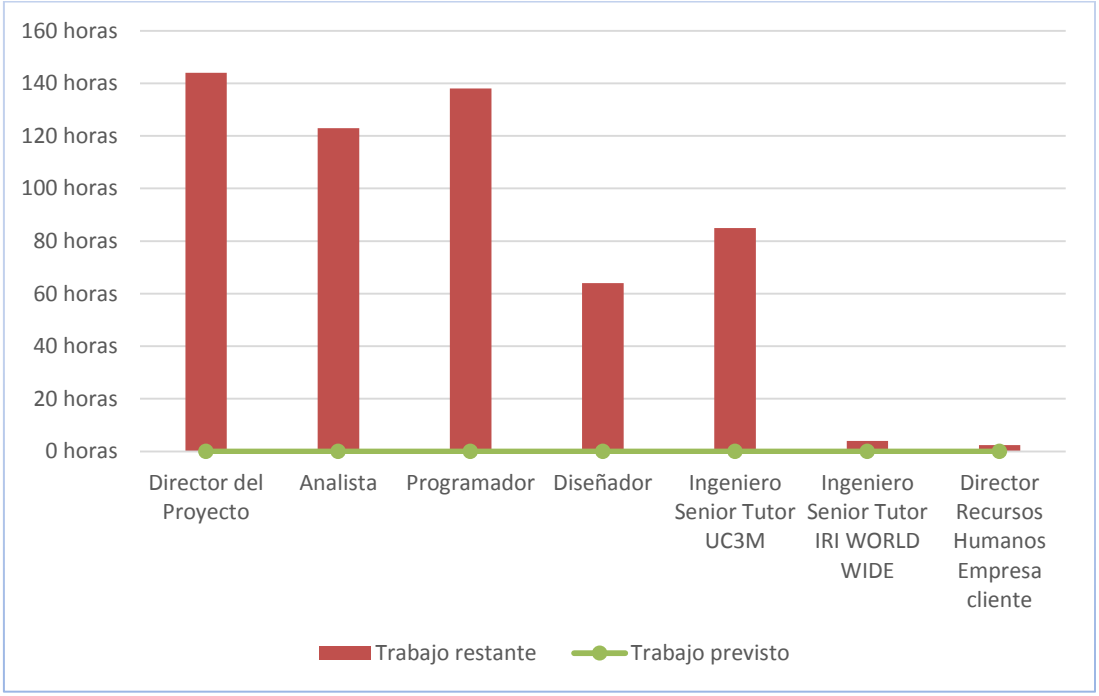


Proyecto: Gantt_tfg
Fecha: jue 14/05/15

Tarea	Resumen del proyecto	Tarea manual	solo el comienzo	Fecha límite
División	Tarea inactiva	solo duración	solo fin	Progreso
Hito	Hito inactivo	Informe de resumen manual	Tareas externas	Progreso manual
Resumen	Resumen inactivo	Resumen manual	Hito externo	

Página 1

Tras el análisis del diagrama de Gantt y la asignación de recursos y tiempo, se obtienen un conjunto de conclusiones. En el siguiente gráfico se observan las horas efectivas trabajadas por cada recurso, que posteriormente servirán para el cómputo final de presupuestos por hora trabajada.



Gráfica 2. Horas por recurso

Una vez calculado el tiempo por recurso, calculamos los costes de estos.

3.2. Presupuestos

3.2.1. Presupuestos parciales

Todos los presupuestos expuestos en las siguientes tablas tienen su correspondiente justificación en el anexo 1: **Tabla de justificación de precios.**

Para la tabla 4: Tabla de presupuestos, recursos humanos, se obtienen los datos según XVII convenio colectivo nacional de empresas de ingeniería y oficinas de estudios técnicos, especificados en la "Resolución de 9 de octubre de 2013, de la Dirección General de Empleo, por la que se registra y publica el XVII Convenio colectivo nacional de empresas de ingeniería y oficinas de estudios técnicos", BOE de 2013. Artículo 33. [10]

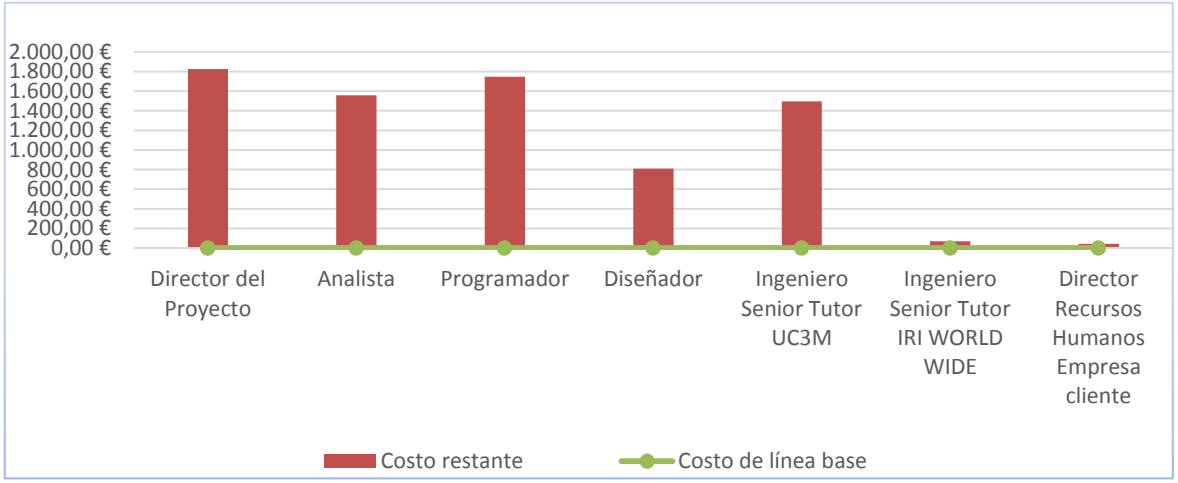
Tabla 3. Presupuestos, recursos materiales

	Concepto	Tiempo de utilización en horas	Coste Total (€)	Vida media de uso (años)	Coste con amortización (€)
Recursos materiales:	PC Dell Optiplex 740	469,00	700	5	36,48 €
	PC Servidor Windows Server 2012	469,00	700	5	36,48 €
	Teclado	469,00	35	1	1,82 €
	Ratón	469,00	15	1	0,78 €
	Total				75,56 €

Tabla 4. Tabla de presupuestos, recursos humanos

	Producto	Horas trabajadas por recurso	Tasa estándar (€/hora)	Coste total €
Recursos humanos:	Director del Proyecto	144,00	12,67	1.824,60 €
	Analista	123,00	12,67	1.558,51 €
	Programador	138,00	12,67	1.748,58 €
	Diseñador	64,00	12,67	810,93 €
	Ingeniero Senior Tutor UC3M	85,00	17,06	1.449,90 €
	Ingeniero Senior Tutor IRI WORLD WIDE	4,00	17,06	68,23 €
	Director Recursos Humanos Empresa cliente	2,40	17,06	40,94 €
	Total			7.501,69 €

Según la tabla anterior podemos graficar los costes de recursos, entendiendo como recursos los diferentes roles e integrantes del proyecto.



Gráfica 3. Coste por recurso

Tabla 5. Tabla presupuestos gasto de licencias

	Concepto	Tiempo de utilización en horas	Coste total (€)	Vida media de uso (años)	Coste con amortización (€)
Gastos de licencias	Licencia de Windows 2007 para el PC de trabajo	469,00	174,17 €	5	9,08 €
	Licencia de Windows server 2012	469,00	259,74 €	5	13,54 €
	Notepad++	469,00	- €	5	- €
	Internet Explorer	469,00	- €	5	- €
	Opera	469,00	- €	5	- €
	Chrome	469,00	- €	5	- €
	Sqlyog	469,00	- €	5	- €
	Licencia PHP, HTML, JavaScript, PHPExcel	469,00	- €	5	- €
	Project professional 2010	469,00	1.369,00 €	5	71,34 €
	Office proffesional plus 2010	469,00	456,66 €	5	23,80 €
	Total				117,75 €

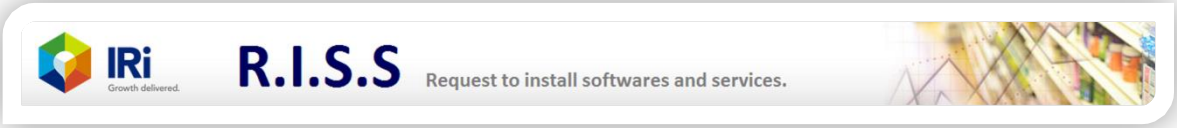
3.2.2. Presupuesto total

Para la ejecución de los presupuestos se suponen ciertas horas calculadas previamente en el diagrama de Gantt. Aunque la previsión se hizo sobre 469 horas trabajadas, en este cómputo se cuentan horas superpuestas, es decir las horas de trabajo del jefe del proyecto y del analista se superponen por lo que cuenta como hora efectiva pero diferenciada en el cálculo de presupuestos. En resumen y sumando el 20% de costes indirectos y el IVA vigente del 21% se resumen los costos en la siguiente tabla.

Tabla 6. Resumen del presupuesto total

Resumen del presupuesto		
Concepto		Importe Total
Recursos materiales		75,56 €
Recursos humanos		7.501,69 €
Gastos de licencias		117,75 €
suma		7.695,00 €
Costes indirectos	20%	1.539,00 €
Total base imponible		9.234,00 €
IVA vigente	21%	1.939,14 €
Total presupuesto		**11.173,14 €**

Capítulo 4: Análisis de la plataforma



4.1. Requisitos

Se describen los diferentes tipos de usuarios que pueden interactuar con el sistema:

- Editor

Engloba dos tipos de usuarios funcionales. Los usuarios editores *helpdesk* y los usuarios editores financieros. Podrán realizar diversas funciones:

- Los usuarios *helpdesk* deberán iniciar sesión en la aplicación.
- Dentro de los usuarios *helpdesk* solo el administrador podrá añadir una cuenta nueva de usuario editor.
- Los usuarios *helpdesk* podrán modificar una petición en espera y ver las aprobadas o rechazadas.
- Los usuarios financieros podrán modificar, aprobar, rechazar o poner en espera una petición.

- Usuario

- Añadir una petición
- Ver el estado de todas sus peticiones
- Consultar guía de usuario.

A continuación se enumeran los requisitos y características del producto.

4.1.1. Requisitos de interfaces externas

Deberá cumplir los siguientes requisitos de interfaces externas:

4.1.1.1. Interfaces de hardware

- Entrada

Será mediante ratón y teclado. El usuario registrará sus peticiones y consultará el estado de las mismas. El usuario *helpdesk* lo utilizará para completar las características y el presupuesto de la petición. Por último, el usuario del departamento financiero para estudiar la petición, aprobarla, rechazarla o ponerla en espera.

- Salida

La salida será a través de un monitor, con una interfaz gráfica *web*.

4.1.1.2. Interfaces de sistema

Cualquier sistema electrónico que soporte HTML5.

4.1.2. Requisitos funcionales

Deberá cumplir los siguientes requisitos funcionales. Se entiende como requisito funcional, la interacción entre el sistema y su ambiente independientemente de su implementación. El ambiente incluye al usuario y cualquier otro sistema externo que interactúa con el sistema [12].

A continuación se presentan los requisitos funcionales del sistema RISS:

RF-RISS-01 El sistema permite añadir una nueva petición.

RF-RISS-02 El sistema permite consultar sus peticiones y comprobar su estado.

RF-RISS-03 El sistema permite a los usuarios consultar en cualquier momento la guía de usuario.

RF-RISS-04 El sistema no permite modificar al usuario su petición tras ser enviada.

RF-RISS-05 El sistema permite a los usuarios *helpdesk* consultar cualquier petición del repositorio, independientemente de su estado, siempre y cuando sea del país del dominio del editor.

RF-RISS-06 El sistema permite a los usuarios *helpdesk* consultar su lista de peticiones pendientes.

RF-RISS-07 El sistema permite a los usuarios *helpdesk* consultar la lista de peticiones aprobadas.

RF-RISS-08 El sistema permite a los usuarios *helpdesk* consultar la lista de peticiones rechazadas.

RF-RISS-09 El sistema permite a los usuarios *helpdesk* consultar su lista de peticiones en espera.

RF-RISS-10 El sistema permite a los usuarios *helpdesk* consultar su lista de peticiones instaladas.

RF-RISS-11 El sistema permite a los usuarios *helpdesk* completar y añadir al repositorio la petición modificada pendiente.

RF-RISS-12 El sistema permite a los usuarios *helpdesk* aprobar la instalación de una petición.

RF-RISS-13 El sistema permite a los usuarios *helpdesk* añadir un fichero de texto, Excel o una imagen a la petición. Será la aprobación financiera. Este archiva se guardará en el servidor y se añadirá al repositorio su directorio.

RF-RISS-14 El sistema permite a los usuarios *helpdesk* añadir un número de ticket de *Infopacc*[1].

RF-RISS-15 El sistema permite a los usuarios *helpdesk* y al usuario financiero consultar las gráficas generadas y actualizadas en cualquier momento.

RF-RISS-16 El usuario *helpdesk* deberá autenticarse para entrar en el sistema.

RF-RISS-17 El usuario *helpdesk* podrá cambiar su contraseña. Esta debe tener como mínimo ocho caracteres.

RF-RISS-18 El usuario *helpdesk* podrá recordar su contraseña, a través del envío de correo electrónico por parte del sistema.

RF-RISS-19 El usuario *helpdesk* podrá cerrar sesión en cualquier momento.

RF-RISS-20 Solo el usuario *helpdesk* administrador podrá crear cuentas de usuario *helpdesk*.

RF-RISS-21 Los nombres de usuario *helpdesk* son únicos. El sistema no permite dos cuentas diferentes con el mismo nombre de usuario.

RF-RISS-22 El sistema actualizará los valores de las gráficas mostradas.

RF-RISS-23 Las gráficas mostradas serán dos gráficos en modo tarta por países y por *software* solicitado respectivamente.

RF-RISS-24 El sistema permite al usuario financiero consultar las peticiones pendientes de aprobación.

RF-RISS-25 El sistema permite al usuario financiero consultar las peticiones aprobadas.

RF-RISS-26 El sistema permite al usuario financiero consultar las peticiones rechazadas.

RF-RISS-27 El sistema permite al usuario financiero consultar los ficheros subidos por el *helpdesk* en cada petición, bien descargando el fichero, si es necesario, o mostrando siempre que se pueda en el navegador (como por ejemplo una imagen).

RF-RISS-28 El sistema permite al usuario financiero exportar en una tabla Excel los datos de la base de datos MySQL.

RF-RISS-29 El sistema permitirá volver a la pantalla principal de cada vista (usuario, *helpdesk* y financiero) en cualquier momento presionando el icono de la aplicación.

RF-RISS-30 En cada modificación de estado de la petición el sistema envía un correo electrónico al usuario solicitante junto con el usuario *helpdesk* o financiero de cada país que corresponda.

4.1.3. Requisitos no funcionales

4.1.3.1. Requisitos del producto

RNF-RISS-01 Existirán tres tipos de usuarios: usuarios solicitantes, *helpdesk* y usuarios financieros. Los dos últimos los englobamos en usuarios editores.

RNF-RISS-02 Dentro de los usuarios *helpdesk*, solo el administrador podrá crear nuevos usuarios *helpdesk* de país.

RNF-RISS-03 El interfaz de usuario se implementará usando HTML5.

RNF-RISS-04 Se utilizarán las tecnologías de PHP, HTML5, MySQL para implementar el sistema.

RNF-RISS-05 Las peticiones de usuario y su modificación por usuarios editores se guardarán inmediatamente.

RNF-RISS-06 Se utilizará siempre la aplicación *web* como plataforma oficial.

RNF-RISS-07 El *software* debe tener una librería de ayuda rápida el cual se activara en la ventana principal de cualquiera de las tres vistas.

RNF-RISS-08 El *software* debe tener un manual que permita al usuario usar la aplicación sin dificultades.

RNF-RISS-09 Las peticiones deberán tener como mínimo un identificador numérico, un nombre de usuario registrado con su correo electrónico, nombre y jefe de departamento, país solicitante, el nombre del *software*, una justificación y el número de licencias de dicho *software*.

RNF-RISS-10 Como mínimo se requiere un elemento de *software* por petición.

RNF-RISS-11 El identificador de las peticiones será único para cada una. Se genera de forma automática y tendrá la forma: "RISS" + "país solicitante" + "id".

RNF-RISS-12 Las justificaciones serán cortas y claras

RNF-RISS-13 Las peticiones se almacenarán en un repositorio. Este será especificado por el equipo de desarrollo.

4.1.3.2. Requisitos de la organización

RNF-RISS-14 El proceso de desarrollo del sistema y los documentos a entregar deberá ajustarse a los estándares de la propia empresa cliente, como la visualización del logo en todo momento o los colores predeterminados de la empresa.

4.1.3.3. Requisitos externos

RNF-RISS-15 El sistema no deberá revelar información personal de los usuarios editores.

RNF-RISS-16 Los usuarios no podrán editar sus propias peticiones.

RNF-RISS-17 Los correos electrónicos enviados por la aplicación no tendrán posibilidad de respuesta.

4.1.4. Especificación de casos de uso

Los casos de uso son un escenario basado en técnicas de UML. Identifican a los actores interactuantes y describen dicha interacción con el programa.

Se han generado varios diagramas de uso para su mejor compresión. En primer lugar, definimos varios actores:

HLP.1. Usuario: Usuario final, puede generar una petición cumpliendo con ciertos requisitos definidos arriba, entre ellos su registro con un nombre de usuario de dominio.

HLP.2. Administrador: Encargado de crear las cuentas de usuario *helpdesk.*

HLP.3. *Helpdesk:* Usuario revisor y modificador de peticiones, deberá añadir el costo y una aprobación financiera en forma de PDF o Word al servidor.

HLP.4. Financiero: Actor que interactúa comprobando y modificando las peticiones, gestionándolas como rechazadas, aprobadas o en espera.

HLP.5. Sistema: conjunto de acciones que realiza la plataforma para que funcione correctamente. Entre ellas las llamadas a la base de datos.

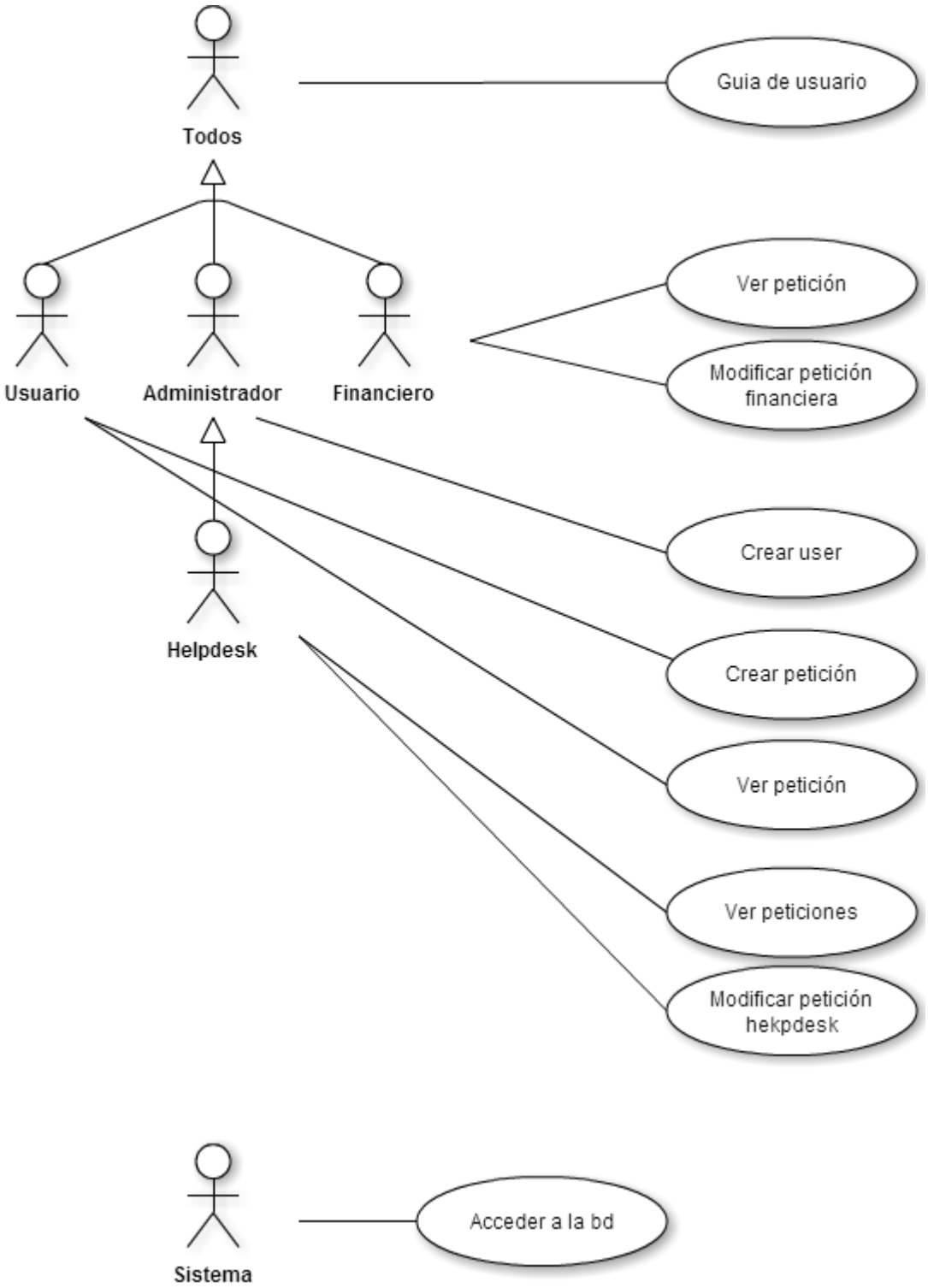


Ilustración 6. Actores

Se definen los casos de uso por tablas con su correspondiente explicación. Los casos de uso especificados han sido creados a través del estándar UML [13].

Identificador: CU-U-01
Nombre: Solicitar nueva petición
Actores: Usuario, base de datos.
Descripción: Crear nueva petición en la base de datos.
Precondición: Tener cuenta de usuario de dominio y correo activado.

Identificador: CU-U-02
Nombre: Ver lista de peticiones de usuario.
Actores: Usuario, base de datos.
Descripción: Listar peticiones solicitadas y ver su estado.
Precondición: Haber solicitado una petición.

Identificador: CU-U-03
Nombre: Ver resumen de petición.
Actores: Usuario, base de datos.
Descripción: Ver el resumen de la petición y su estado actual dentro del flujo del sistema.
Precondición: Haber solicitado una petición.

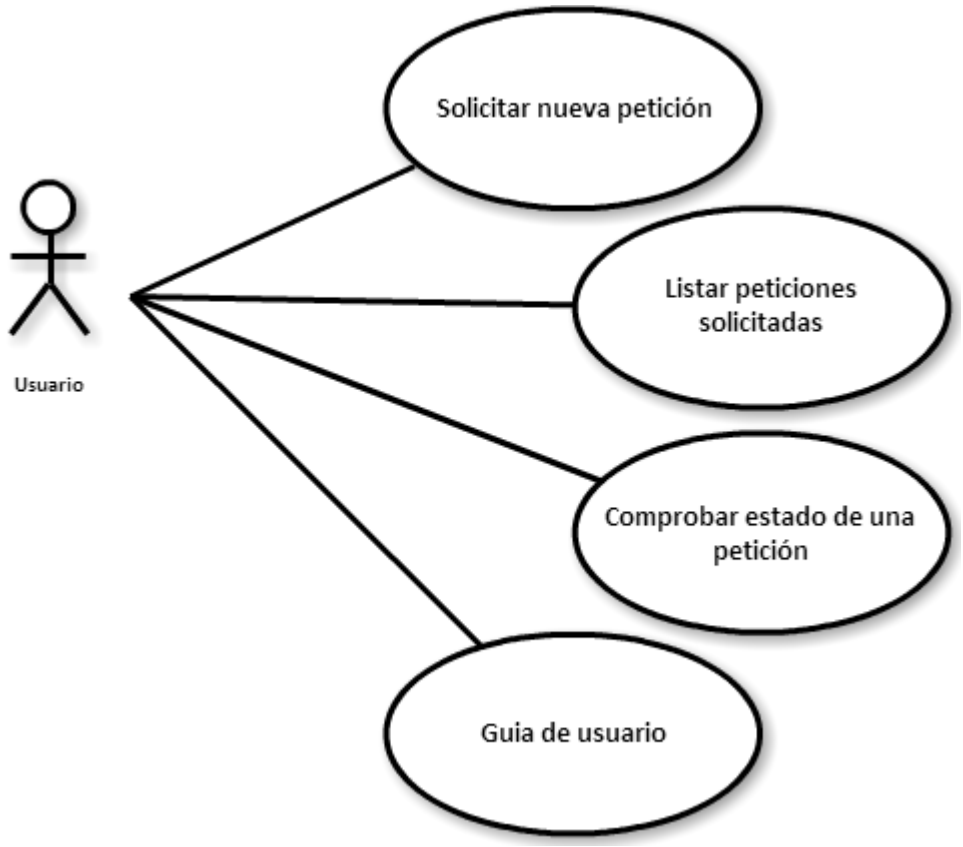


Ilustración 7. Caso de uso de usuario

Identificador: CU-A-01
Nombre: Crear cuenta de usuario *helpdesk* nueva.
Actores: Administrador, base de datos.
Descripción: Crear cuenta de usuario *helpdesk* nueva. Será única por país.
Precondición: Inicio de sesión como admin.

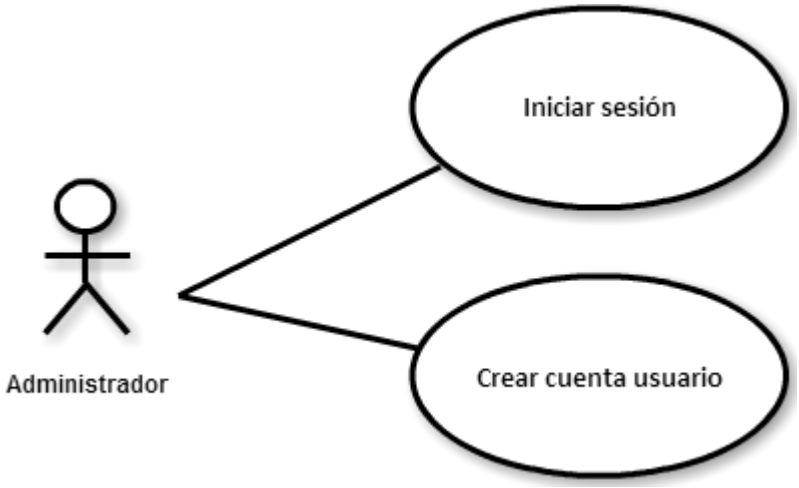


Ilustración 8. Caso de uso de administrador

Identificador: CU-H-01
Nombre: Listar peticiones pendientes
Actores: *Helpdesk*, base de datos.
Descripción: Se listaran todas las peticiones solicitadas por los usuarios que aún están pendientes. Serán ordenadas por fecha. El usuario *helpdesk* solo podrá ver las propias de su país.
Precondición: Inicio de sesión como usuario *helpdesk.*

Identificador: CU-H-02
Nombre: Listar peticiones aprobadas.
Actores: *Helpdesk*, base de datos.
Descripción: Se listaran todas las peticiones aprobadas por el usuario financiero. Serán ordenadas por fecha. El usuario *helpdesk* solo podrá ver las propias de su país.
Precondición: Inicio de sesión como usuario helpdesk. Usuario financiero haya aprobado alguna petición.

Identificador: CU-H-03
Nombre: Listar peticiones rechazadas.
Actores: *Helpdesk*, base de datos.
Descripción: Se listaran todas las peticiones rechazadas por el usuario financiero. Serán ordenadas por fecha. El usuario *helpdesk* solo podrá ver las propias de su país.
Precondición: Inicio de sesión como usuario *helpdesk.* Usuario financiero haya rechazado alguna petición.

Identificador: CU-H-04
Nombre: Modificar petición solicitada.
Actores: *Helpdesk*, base de datos.
Descripción: Selección de petición solicitada por el usuario y modificación de ciertos campos obligatorios. Se guardarán los cambios en la base de datos del servidor.
Precondición: Inicio de sesión como usuario *helpdesk.* Selección de la lista de peticiones solicitadas.

Identificador: CU-H-05
Nombre: Modificar petición en espera.
Actores: *Helpdesk*, base de datos.

Descripción: Selección de petición en espera usuario y modificación de ciertos campos obligatorios. Se guardarán los cambios en la base de datos del servidor.
Precondición: Inicio de sesión como usuario *helpdesk.* Selección de la lista de peticiones en espera. El usuario financiero rellena el campo de notas, explicando los campos a modificar.

Identificador: CU-H-06
Nombre: Buscar una petición por NUMRISS.
Actores: *Helpdesk*, base de datos.
Descripción: Dentro del cuadro de búsqueda se inserta el NUMRISS a buscar. Aunque no pertenezca al propio país del *helpdesk*, esta podrá ser vista (sin posibilidad de modificación).
Precondición: Inicio de sesión como usuario *helpdesk.*

Identificador: CU-H-07
Nombre: Visualizar gráficos.
Actores: *Helpdesk*, base de datos.
Descripción: Se podrán visualizar los gráficos generados de forma automática, descargarlos y guardarlos en formato PNG, JPG, PDF o SVG.
Precondición: Inicio de sesión como usuario *helpdesk.*

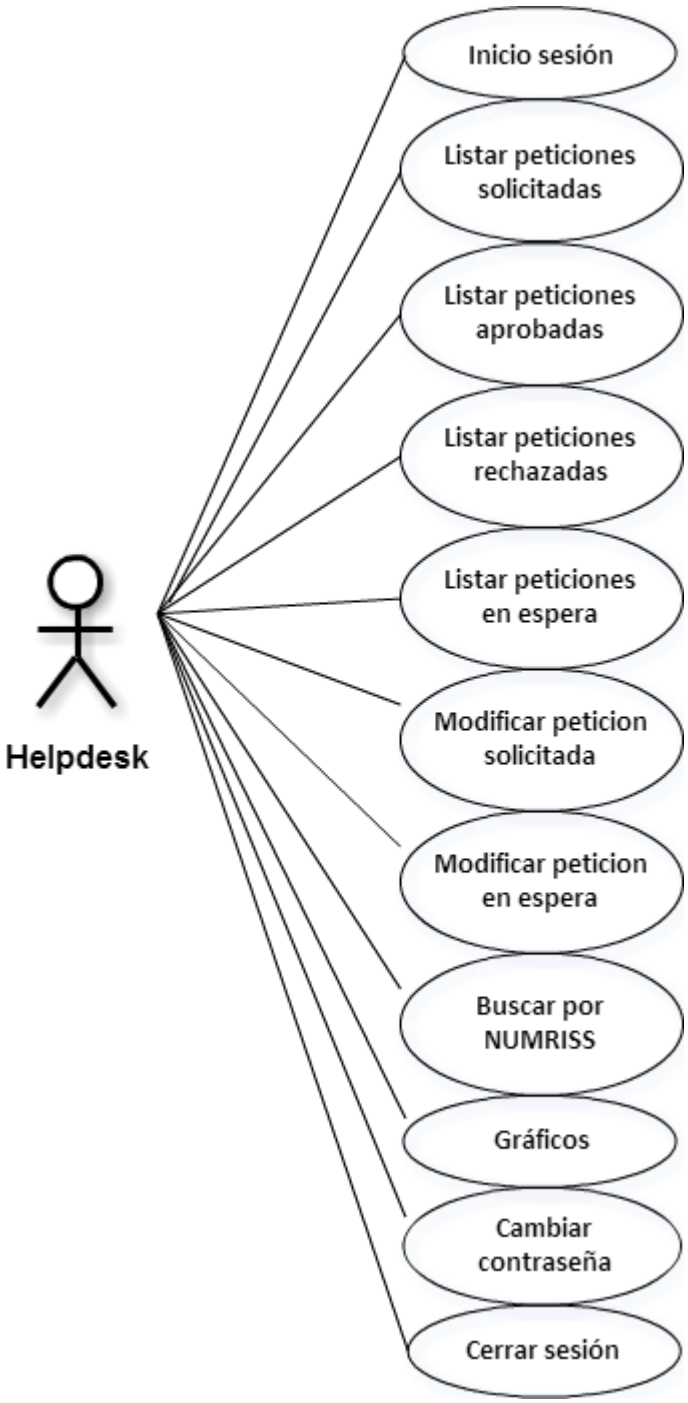


Ilustración 9. Caso de uso usuario *Helpdesk*

Identificador: CU-F-01
Nombre: Listar peticiones pendientes
Actores: Financiero, base de datos.
Descripción: Se listaran todas las peticiones solicitadas por los usuarios y modificadas por el *helpdesk* que aún están pendientes. Serán ordenadas por fecha.
Precondición: Tener permisos de usuario financiero.

Identificador: CU-F-02
Nombre: Listar peticiones aprobadas.
Actores: Financiero, base de datos.
Descripción: Se listaran todas las peticiones aprobadas anteriormente.

Precondición: Tener permisos de usuario financiero. Usuario financiero haya aprobado alguna petición.

Identificador: CU-F-03
Nombre: Listar peticiones rechazadas.
Actores: Financiero, base de datos.
Descripción: Se listaran todas las peticiones rechazadas anteriormente.
Precondición: Tener permisos de usuario financiero. Usuario financiero haya rechazado alguna petición.

Identificador: CU-F-04
Nombre: Modificar petición pendiente.
Actores: Financiero, base de datos.
Descripción: Selección de petición modificada por el usuario *helpdesk* y modificación de ciertos campos obligatorios. Se guardarán los cambios en la base de datos del servidor.
Precondición: Tener permisos de usuario financiero. Selección de la lista de peticiones solicitadas.

Identificador: CU-F-05
Nombre: Visualizar gráficos.
Actores: Financiero, base de datos.
Descripción: Se podrán visualizar los gráficos generados de forma automática, descargarlos y guardarlos en formato PNG, JPG, PDF o SVG.
Precondición: Tener permisos de usuario financiero.

Identificador: CU-F-06
Nombre: Exportación a Excel.
Actores: Financiero, base de datos.
Descripción: Exportación de los campos relevantes y permitidos de la base de datos a una tabla de Excel.
Precondición: Tener permisos de usuario financiero.

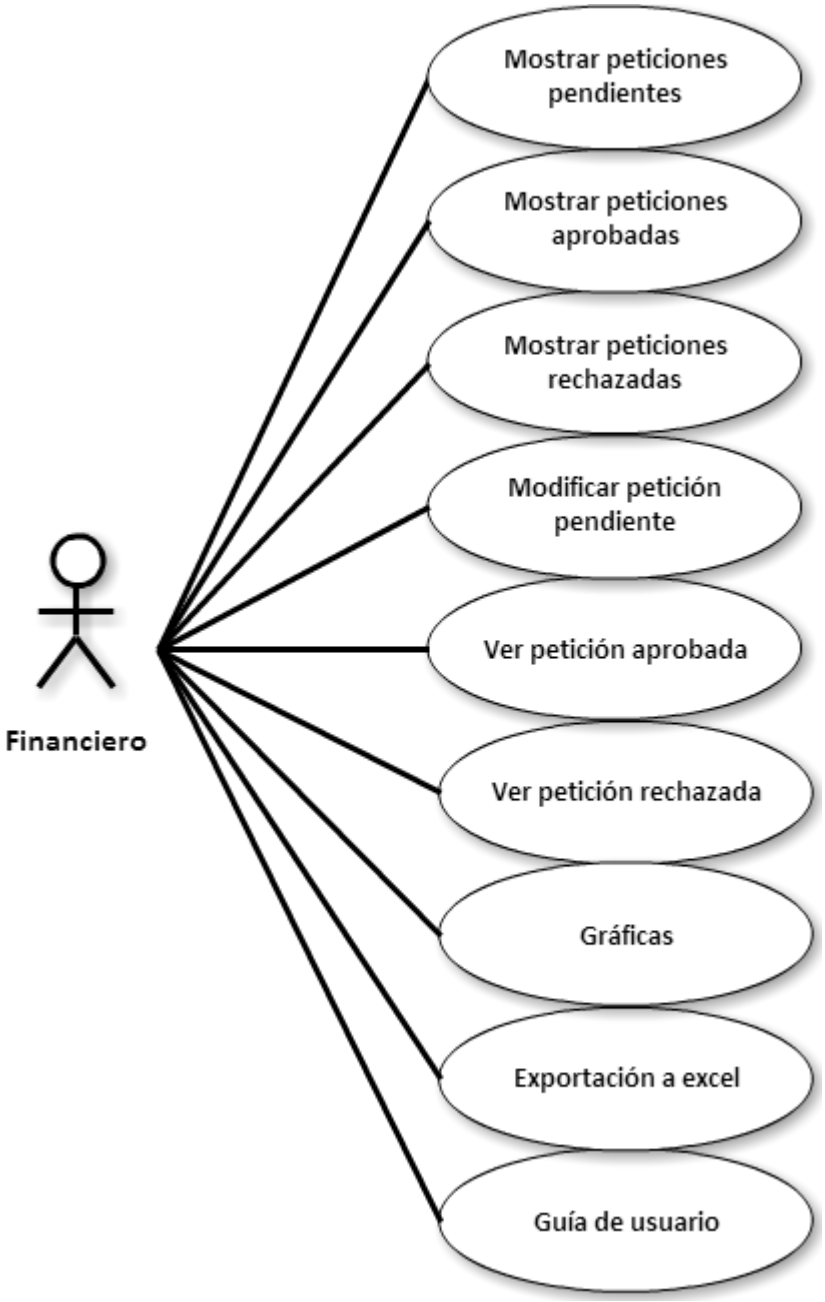


Ilustración 10. Caso de uso usuario financiero.

4.2. Arquitectura del sistema

La arquitectura de una aplicación es la vista conceptual de la estructura de esta. Toda aplicación contiene código de presentación, código de negocio o procesamiento y código de almacenamiento de datos. La arquitectura de las aplicaciones difiere según cómo y dónde está distribuido este código.

4.2.1. Arquitectura cliente/servidor

La plataforma RISS, es una aplicación *web* que cumple la arquitectura de cliente/servidor utilizada en un Sistema distribuido, entendiendo como sistema distribuido: "sistema informático compuesto por un conjunto de nodos de procesamiento comunicados y coordinados mediante una red que permite el intercambio de mensajes entre los mismos". [9]

Usando el protocolo HTTP, los usuarios de la plataforma podrán acceder desde cualquier dispositivo compatible con HTML5 y conectado a internet a la aplicación. Esta arquitectura está instalada en un servidor externo Windows Server 2012 [10]. Solo a través de la aplicación será posible la extracción o modificación de los datos.

En la siguiente ilustración se observa el modelo seguido de cliente/servidor:

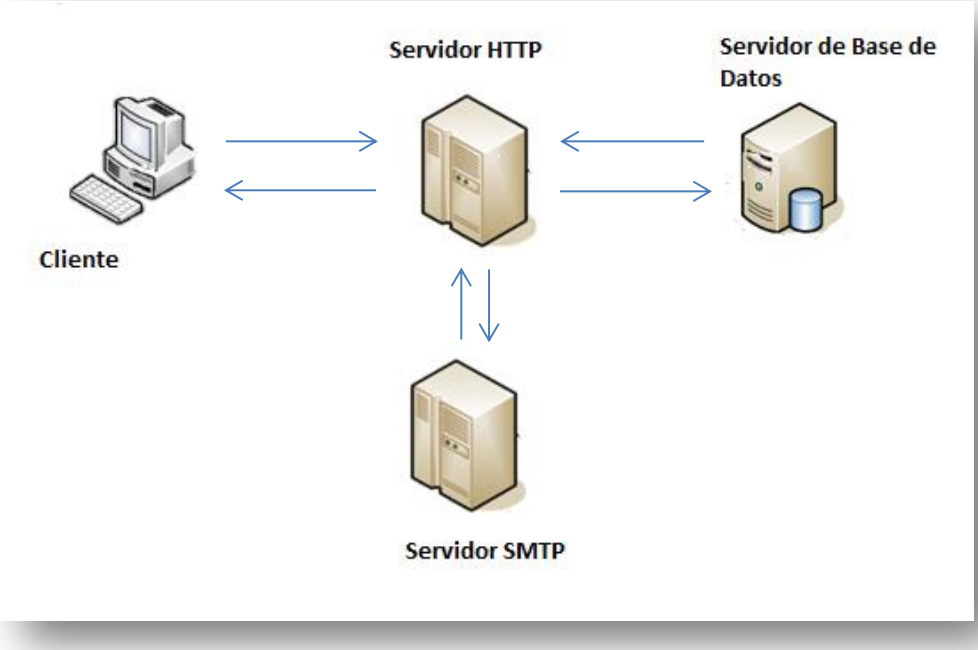


Ilustración 11.Modelo cliente servidor

Aunque es una arquitectura de estructurado habitual, esta etapa se afrontó justo después del análisis de los requisitos, ya que vendrá implícitamente impuesta por estos. El desarrollo de este proyecto se realiza sobre una infraestructura definida, con unos equipamientos y un *software* previos a la propia aplicación.

El modelo cliente/servidor utilizado distribuye la aplicación entre dos o más componentes, definiendo, en este caso, el cliente y los servidores que se ocupan de diferentes responsabilidades. Este diseño define un modelo de interacción basado en el diálogo petición-respuesta.

El cliente, el encargado de iniciar el diálogo, envía una petición que recoge la aplicación *web*. Está diseñado para que soporte la interacción con el usuario final, asumiendo que cada petición tendrá una respuesta, incluyendo respuestas de error. Las características del cliente son entre otras:

- No comparten los recursos con otros clientes
- No requieren hardware muy potente (siempre que soporte HTML5)
- No se comunican con otros clientes (clientes puros)
- Pueden tener direcciones IP dinámicas
- Facilidad de uso y seguridad serán sus principales pilares.

La petición es recogida y procesada por el servidor quien presta el servicio y responde a las peticiones recibidas en forma de datos. Gestiona y comparte sus recursos con varios clientes a los que sirve. Se destacan entre las características del servidor:

- La necesidad de estar encendido para el uso de la plataforma
- Dirección IP estática.
- Se utilizan dos servidores principales, uno encargado del correo electrónico, donde se implementa el protocolo SMTP y otro de almacenamiento de datos.

Por tanto, la arquitectura distribuida de la plataforma es tal que la gestión de los datos está en manos del servidor y la presentación de la interfaz y el resultado de dicha petición en el lado del cliente.

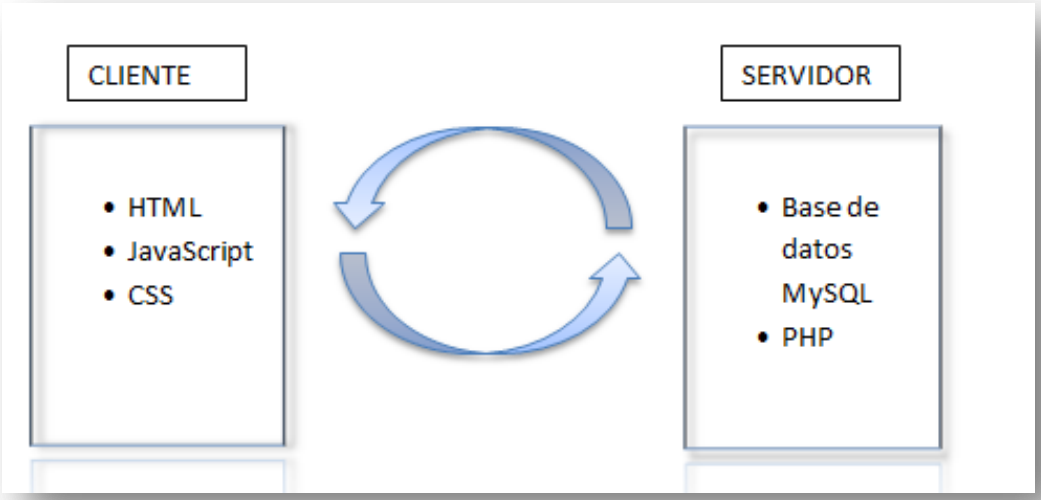


Ilustración 12. Estructura básica cliente/servidor

La capa de presentación o interfaz gráfica corresponde a la aplicación *web* que interacciona con el usuario, presenta y recibe los datos a través de entradas de respuestas de peticiones, corresponde con el cliente. La capa de aplicación o negocio, la lógica de la plataforma se implementa en el servidor, responsable este de las tareas propias de las reglas de la aplicación sobre los datos y las entradas de usuarios. Por último la capa de datos donde se encuentra el almacenamiento y el acceso a los mismos, también es responsable la parte del servidor.

Una vez realizada la petición, se procesa en lenguaje PHP y genera respuestas.

En el siguiente diagrama se especifica cada capa de la arquitectura y su procedencia, si es responsabilidad del cliente o del servidor.

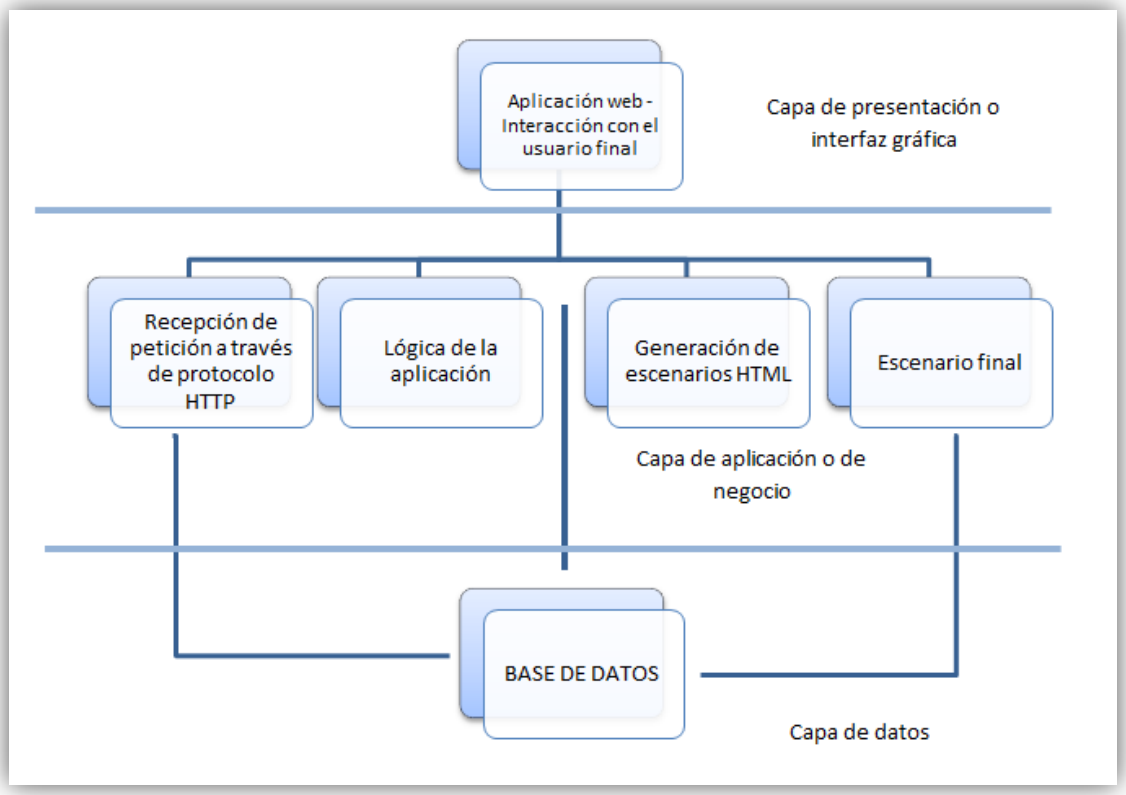


Ilustración 13. Diagrama por capas de arquitectura

4.2.2. Capa de presentación

El cliente genera una petición, como por ejemplo mostrar una petición guardada en la base de datos. Para ello deberá interaccionar con la aplicación *web*. Esta se muestra a través de formularios o pantallas generadas con HTML, las vistas (MVC, detallado en el apartado de Diseño).

La petición se envía a través del protocolo HTTP, con métodos como GET, POST, HEAD... la cual será recibida por el servidor, encargado de gestionarla.

Dentro de las funciones de esta capa, se encuentran la administración de la interfaz de la aplicación, realización de validaciones locales, generar solicitudes a la base de datos o recibir las respuestas del servidor.

Para la interfaz *web,* se utilizó HTML5 para la creación de formularios y pantallas de la plataforma, JavaScript para la interacción de ciertos botones, que como ya se ha detallado arriba gestiona la buena presentación de la página. Por último, CSS, que gestiona una hoja de estilos común para la estilización de las páginas.

4.2.3. Capa de negocio

La capa de negocio, núcleo principal de la plataforma, es quien gestiona la ejecución del sistema. Pertenece al ámbito del servidor, por lo que es la encargada de procesar las peticiones HTTP recibidas e interaccionar si es necesario, con la capa de datos. Por tanto es quien realiza todas las funciones hasta la creación de los escenarios finales.

Este módulo será el que acceda a la base de datos, como se representa en el esquema anterior, con el fin de recuperar los datos y mostrarlos en los escenarios finales. Por ejemplo, en el caso anterior de mostrar una petición guardada en la base de datos, este módulo accedería a la capa de datos, recolecta la información necesaria, crea el escenario de presentación y su interfaz final, la cual envía a la capa anterior, la capa de presentación.

La lógica de la aplicación se desarrolla a través de PHP, PHPExcel y el API de **Highcharts.** [13] Este módulo actúa como controlador (MVC, detallado en el apartado de Diseño).

4.2.4. Capa de datos

Esta capa engloba todas las tablas creadas en el servidor de bases de datos. Contiene la información tanto de los usuarios, como de las peticiones creadas y su estado. Desde la capa de negocio, se recupera esta información de forma ordenada creado la respuesta final a la petición del cliente y mostrándola por pantalla.

La base de datos es una base MySQL. Las tablas relacionadas entre sí y con claves primarias se especifican en el apartado de Diseño.

Por último y para resumir, en la arquitectura cliente/servidor no es estática e inflexible. En el caso de la plataforma RISS, se utiliza un servidor extra, el servidor de correo o SMTP. Como se detalla en el capítulo de Implementación, este servidor es utilizado por la lógica de la aplicación para el envío y recepción de correos electrónicos de la aplicación. El único requisito será la necesidad de tener una cuenta de correo de dominio de la empresa.

4.3. Diseño

El propósito de este punto es dar una visión detallada de cómo funciona el sistema. El objetivo de diseño debe cumplir con los requisitos dados por la empresa cliente.

4.3.1. Descripción básica del sistema

La plataforma se constituye con un *software* desarrollado de forma propia. Se siguen unos modelos y patrones para su diseño, que se especifican a continuación.

La arquitectura del *software* RISS sigue el patrón Modelo-Vista-Controlador. La elección de este patrón radica en las siguientes ventajas:

· Facilidad de implementación en aplicaciones con interfaz gráfica dada la separación de los componentes.

· Facilidad de reutilización de componentes.

· Facilidad para realizar pruebas unitarias de los componentes.

Los diferentes componentes del sistema son los que se muestran en la figura.

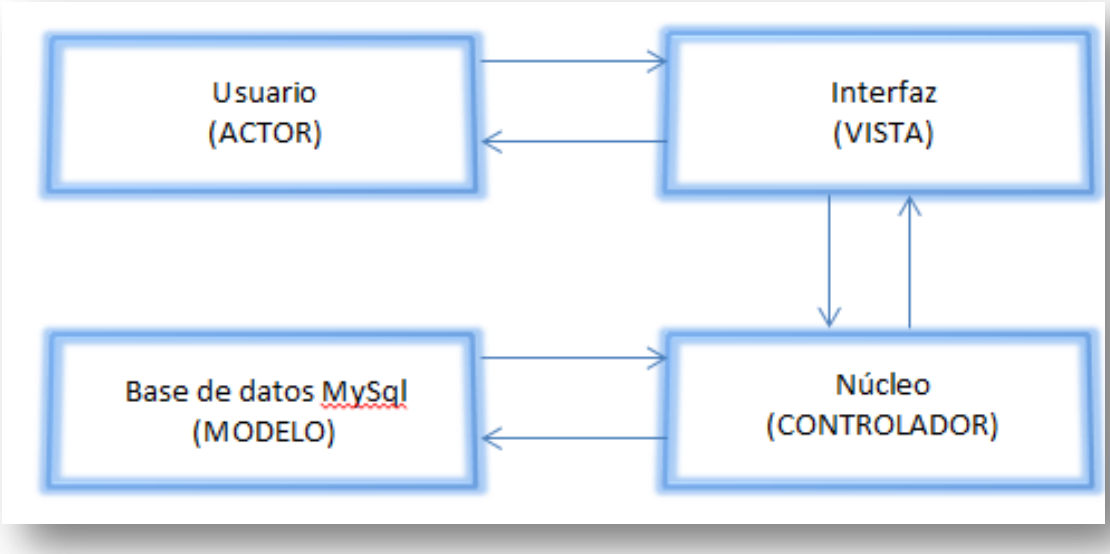


Figura 14. Componentes del patrón: modelo-vista-controlador (MVC).

El usuario o actor, quien interactúa con la aplicación a través de la interfaz (vista). Encargada de presentar la información de la aplicación al usuario y de dar la posibilidad de interactuar. Controlador, este componente gestiona las interacciones del usuario con la

aplicación. Por último el modelo quien determina el comportamiento, las funcionalidades, características y restricciones.

4.3.2. Diagrama de flujo de la aplicación

Los modelos de flujo de datos ayudan al análisis del sistema, mostrando la forma en la que los datos se asocian entre sí y como se procesan los datos en el sistema.

Se puede observar a continuación el diagrama de flujo de la plataforma RISS, representado mediante el estándar UML [13].

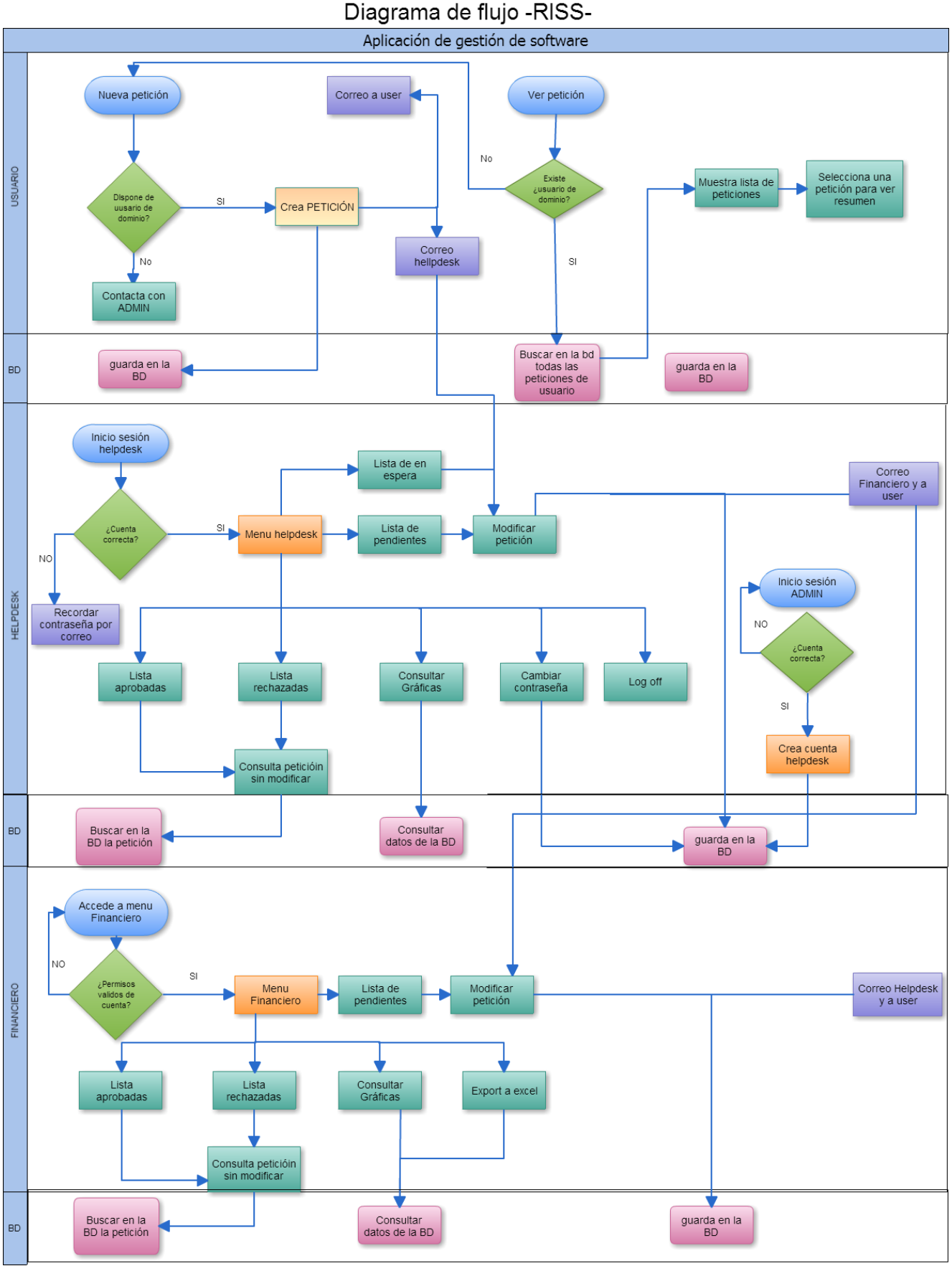


Ilustración 15. Diagrama de flujo RISS

4.3.3. Diseño detallado de la aplicación

En la siguiente figura podemos observar la distribución de la plataforma, dividida en tres partes diferenciadas.

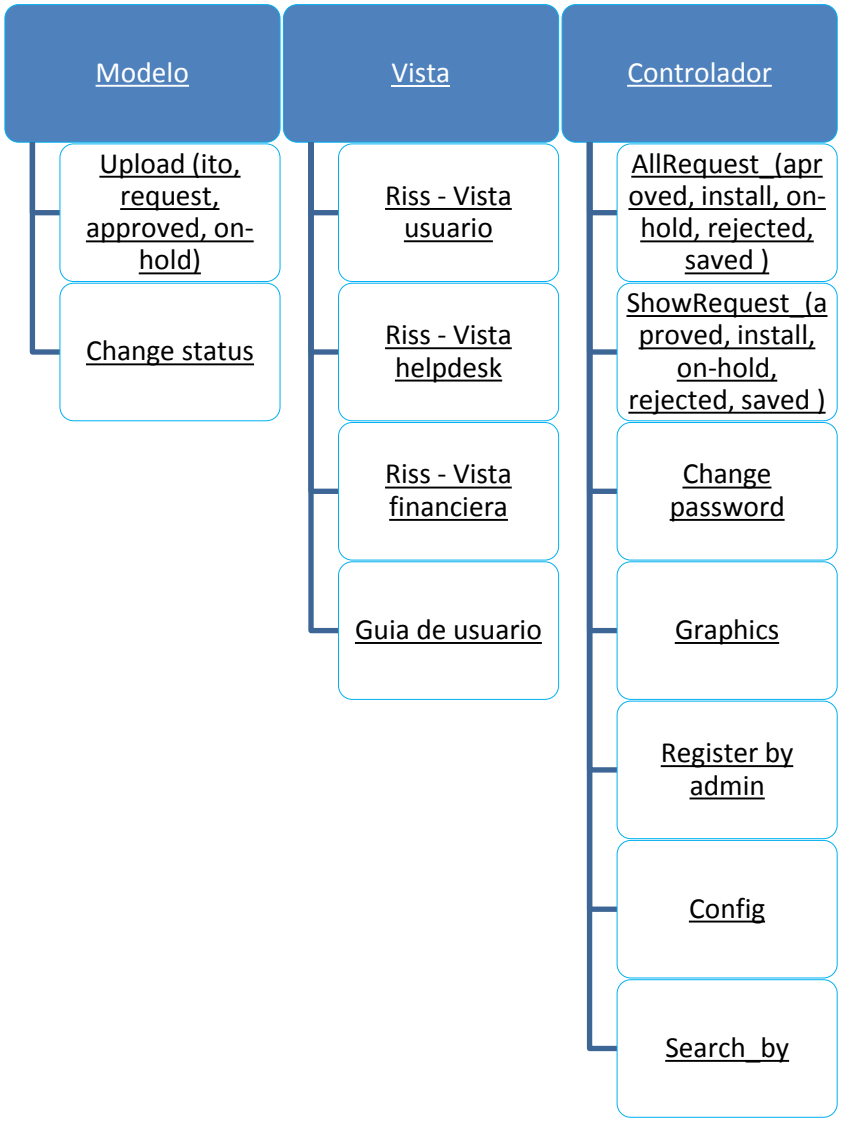


Figura 16. Gráfico MVC-RISS

Se describen con detalle las clases más significativas:

- **Upload**: Encargada de guardar las modificaciones realizadas por el usuario final en una petición. Se hará distinción si es una petición aprobada, rellenada por el usuario *helpdesk*, recién subida por el usuario, o puesta en espera. Es el nexo de unión entre la interfaz de salida y la base de datos determinada. **Pertenece al paquete de modelo.**
- **Change status:** Según el flujo normal de la petición, esta puede estar en diferentes estados. De entre ellos están: "Submitted to heldesk"," Submitted for approval",

"Approved"," Rejected", "OnHold" o " Installed". Según su estado se controla que usuarios pueden acceder a ellas. **Pertenece al paquete de modelo.**

- **AllRequestSaved_:** Muestra una lista de peticiones dependiendo de su estado actual. Cabe destacar, que las peticiones de usuarios *helpdesk* se filtran también por país. ***Pertenece al paquete controlador.***
- **ShowRequest_:** Encargada de representar la información proporcionada por el sistema y de obtener la información que el usuario introduce. Es el nexo de unión entre el usuario y la base de datos. Muestra peticiones aprobadas, instaladas, en espera o guardadas por el usuario y listas para rellenar. ***Pertenece al paquete controlador.***
- **Change password:** Permite el cambio de contraseña al usuario *helpdesk*. Esta contraseña se guardará de forma encriptada en la base de datos. Solo podrá acceder a ella si accede a la opción recordar contraseña via correo electrónico. Las contraseñas se encriptarán con formato MD5. ***Pertenece al paquete controlador.***
- **Graphics**: Contiene las estadísticas de uso mostradas en graficos a través de la API de **Highcharts.** Contiene los métodos necesarios para desencapsular su información. ***Pertenece al paquete controlador.***
- **Config:** Contiene la información necesaria para la conexión a la base de datos. El nombre del formulario, el usuario host y su contraseña. ***Pertenece al paquete controlador.***
- **Vistas**: Cada una pertenece a la vista principal de un usuario determinado, donde podrá acceder a sus opciones según privilegios. *Existen tres tipos de usuario y por tanto de vistas, usuario, helpdesk y financiero****. Pertenece al paquete vista.***

4.3.4. Diseño de la base de datos.

El diseño de la base de datos del proyecto se ha realizado en MySQL, utilizando un modelo con distintas tablas, relacionadas por el campo id.

Tabla de petición general:

Campo	Descripción	Tipo
id	Id	int(100) NOT NULL
date	Fecha de creación de la petición	date NOT NULL
date_helpdesk	Fecha de modificación de *helpdesk*	date
date_euoito_onhold	Fecha de modificación de financiero (si en espera)	date
date_euoito	Fecha de modificación de financiero	date
numriss	Numriss (Identificador "RISS+PAÍS+ID")	varchar(100)
username	Nombre de usuario	varchar(200) NOT NULL
userdpto	Nombre de departamento	varchar(200) NOT NULL
managername	Nombre del manager del departamento	varchar(200) NOT NULL
country	País	varchar(200) NOT NULL
status	Estado	int(1) NOT NULL
note_euoito	Notas del departamento financiero	tinytext
purchase_require	¿Compra necesaria?	varchar(100)
file_name	Nombre del directorio del fichero	varchar(800)
install	Instalación inmediata	tinytext
infopacc_ticket	Numero de ticket de "*Infopack*"	text

Tabla 7. Tabla de la BD: petición

Tabla de *helpdesk*:

Campo	Descripción	Tipo
id	Id	int(200) NOT NULL
hostname	Nombre del dispositivo(s) donde instalar	varchar(200) NOT NULL
model	Modelo del dispositivo	varchar(200) NOT NULL
incremental	¿Es un PC nuevo?	varchar(5) NOT NULL

financial	¿Necesita aprobación financiera?	varchar(5) NOT NULL
upgradehw	¿Necesita actualización de hardware?	varchar(5) NOT NULL

Tabla 8. Tabla de la BD helpdesk

Tabla de *software*:

Campo	Descripción	Tipo
id	Id	int(100) NOT NULL
unic	Identificador de *software*	int(100) NOT NULL
swname	Nombre del *software*	varchar(200) NOT NULL
temuse	¿Es de uso temporal?	varchar(5) NOT NULL
fechauso	Si es de uso temporal, ¿Cuál es la fecha máxima?	varchar(200) NOT NULL
justification	Justificación	tinytext NOT NULL
amount	Cantidad de *software*	int(200) NOT NULL
la	Nombre de licencia	varchar(200)
freeshare	¿*Software* compartido?	varchar(2)
ownedIRI	Número de licencias ya compradas	varchar(15)
licowned	Número de licencias en uso	varchar(15)
licasis	Número de licencias por comprar	varchar(15)
cost	Coste	varchar(200)
notes	Notas	tinytext

Tabla 9. Tabla de la BD de *software*

Tabla de inicio de sesión:

Campo	Descripción	Tipo
id	Id	int(11) NOT NULL
username	Nombre de usuario *helpdesk*	varchar(500) NOT NULL
password	Contraseña	varchar(500) NOT NULL
country	País al que pertenece	varchar(5) NOT NULL

Tabla 10. Tabla de la BD de inicio de sesión

Todas las tablas tienen como claves primarias el id. Además la tabla de peticiones generales tendrá como identificador de petición "numriss" compuesto de la siguiente forma: "RISS+país+id". La tabla de *software* se referencia con el id al igual que las demás, pero

como una misma petición puede solicitar varios elementos de *software,* el id deja de ser único, pasando a ser su clave el identificador de *software*.

4.3.5. Tablas entidad-relación

A continuación se muestra de forma gráfica la tabla de entidad relación de la base de datos. Se reflejan tanto las relaciones entre diferentes tablas como sus claves.

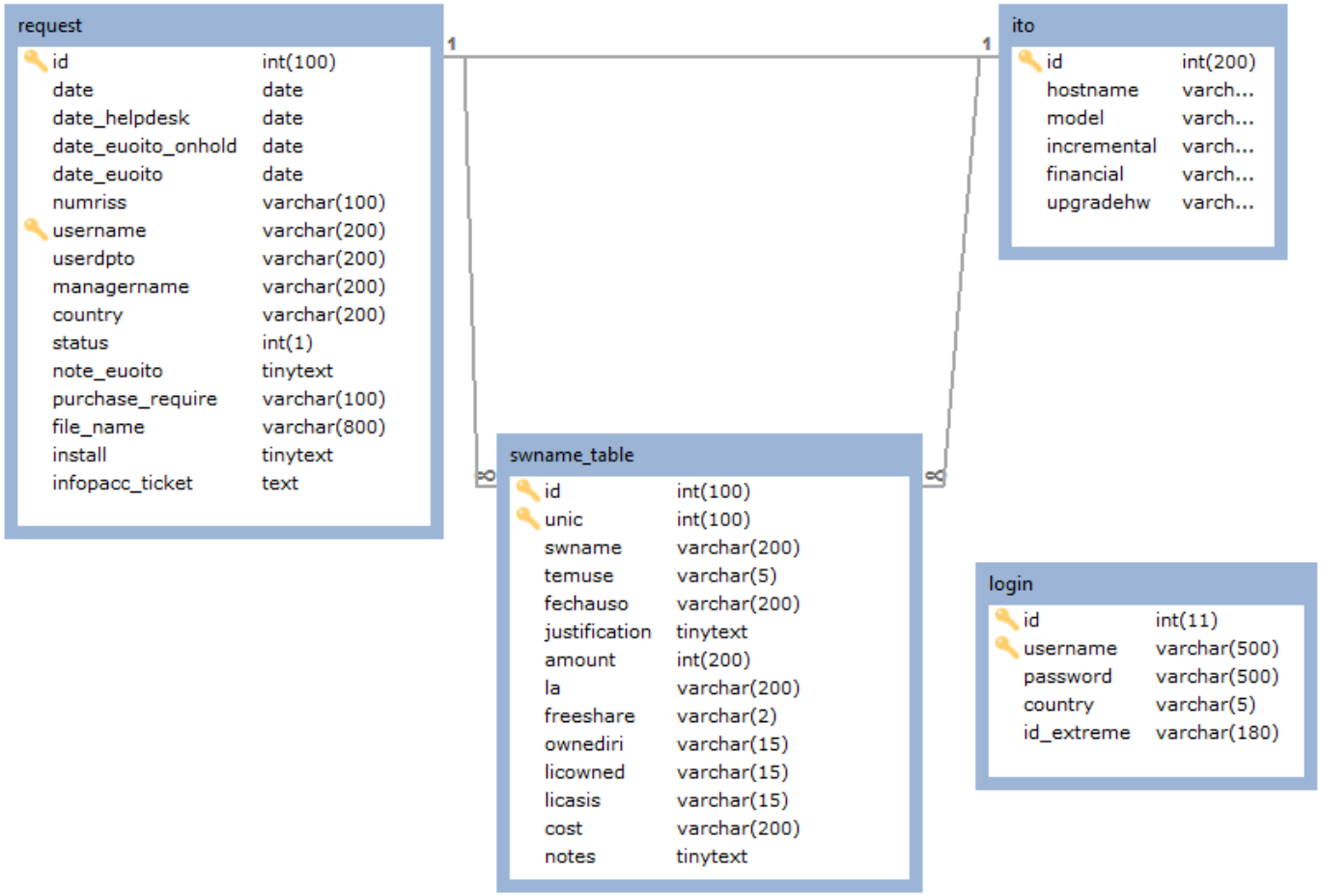


Tabla 11. Tabla entidad-relación de la base de datos completa.

4.4. Implementación

El propósito de este capítulo es dar una visión detallada de cómo funciona el sistema implementado y ya en funcionamiento.

Se incluye la información necesaria, aunque no el código completo del sistema. Solo se explicaran los aspectos más relevantes de implementación. Más adelante se incluye en este documento el plan de pruebas, donde se verifica la viabilidad de la plataforma. El objetivo de diseño debe cumplir con los requisitos de la empresa, ya mencionados, sin embargo es necesario contar con requerimientos específicos para su implementación.

4.4.1. Datos generales de implementación

La aplicación tiene una estructura sólida pero flexible cuyo objetivo es facilitar posibles cambios de diseño y funcionalidad requeridos por el cliente. Asimismo, se hará una exhaustiva depuración para asegurar la limpieza de potenciales errores. Para que la experiencia del usuario final sea placentera, se diseña una interfaz sencilla e intuitiva, teniendo como *frontend* una interfaz gráfica *web* cuidada y estilizada. Se utiliza para la interfaz de usuario una aplicación web dirigida por eventos, desarrollada con HTML5, la cual se basa en que la sucesión de cualquier tipo de acción viene precedida de un evento que ha desencadenado el usuario previamente. Por ejemplo pulsación de teclado o ratón.

El *backend* de la plataforma se generó con módulos PHP. Cabe destacar que dentro de los requisitos del cliente, se encuentra la especificación de poder descargar los datos y exportarlos a Excel (**RF-RISS-28**). Para ello se utiliza el módulo PHPExcel.

La instalación de la base de datos y del proyecto se realizó en un servidor Windows Server 2012 [14] determinado por la empresa cliente. El diseño y la gestión de los datos de la aplicación se hicieron a través de unas bases de datos de MySQL.

Para su prueba en local, se utilizan el servidor Apache y MySQL de nuevo.

Dentro del proyecto, destacan varios puntos principales de la implementación:

- Generación de gráficas
- Acceso a la base de datos.
- Envío de correos electrónicos

4.4.2. Generación de gráficas

Para la solventar el requisito de la generación de gráficas (**RF-RISS-15**) se utiliza el API de **Highcharts.** [8] **.**Funciona tanto en todos los navegadores móviles como de escritorio, como en iPhone o iPad. La licencia es gratuita y solo necesita dos archivos JavaScript para ejecutar: el núcleo highcharts.js y, o bien jQuery, MooTools o marco Prototype.

Para utilizar esta API únicamente se debe incluir la librería JavaScript y definir una nueva instancia Higthcharts. En el siguiente fragmento de código se puede ver el mecanismo para comenzar a utilizar los gráficos.

```
<script src="../js/jquery-2.1.1.min.js"></script>
<script src="http://code.highcharts.com/highcharts.js"></script>
<script src="http://code.highcharts.com/highcharts-3d.js"></script>
<script src="http://code.highcharts.com/modules/exporting.js"></script>
<div id="container" style="min-width: 310px; height: 400px; max-width: 600px; margin: 0 auto; align: "></div>
<div id="container2" style="min-width: 310px; height: 400px; max-width: 600px; margin: 0 auto; align: "></div>
```

Ilustración 17. Definir instancia Higthchart

A continuación se muestra el ejemplo de la configuración de una instancia de JavaScript:

```
script type="text/javascript">
(function () {
    $('#container').highcharts({
        chart: {
            type: 'pie',
            options3d: {
                enabled: true,
                alpha: 45
            }
        },
        title: {
            text: 'Statistics of requests per countries '
        },
        subtitle: {
            text: 'RISS Statistics'
        },
        plotOptions: {
            pie: {
                innerSize: 100,
                depth: 45
            }
        },
        series: [{
            type: 'pie',
            name: '#requests per country',
            innerSize: '50%',
            data: [
                ['Spain',   <?PHP echo $SP; ?>],
                ['France',   <?PHP echo $FR; ?>],
                ['Italy',       <?PHP echo $IT; ?>],
                ['Netherlands', <?PHP echo $NL; ?>],
                ['Germany',    <?PHP echo $DE; ?>],
                ['UK',     <?PHP echo $UK; ?>],
                ['Grecee ACE',     <?PHP echo $GRA; ?>],
                ['Grecee IRI',     <?PHP echo $GRI; ?>],
                ['Litmus',     <?PHP echo $LT; ?>],
                ['Lentus',     <?PHP echo $SE; ?>],
                ['Sales Out',     <?PHP echo $SO; ?>],
                {
                    name: 'Others',
                    y: 0.7,
                    dataLabels: {
```

Ilustración 18. Código de una gráfica en modo tarta 3D

4.4.3. Acceso a la base de datos

En este apartado se muestra cómo se almacenan variables en el fichero de configuración de la aplicación.

Este código se incluye en el resto de ficheros PHP. Como se ve en la imagen comentada, el primer apartado son los campos para la conexión de la base de datos. El siguiente apartado será el inicio de la sesión para un usuario.

```php
<?php
/* Campos de la variable conexion a la bd */

$host = 'localhost';
$usernamedb = 'xxxxx';
$password = 'xxxxx';
$dbname = 'riss';
```

Ilustración 19. config.php

En el documento se detalla cómo la aplicación está diseñada de forma segura. La privacidad de los usuarios es un punto importante en toda la plataforma. El uso de esta aplicación no puede ser fuente de vías de escape en la seguridad de la empresa.

Se utilizó el método MD5 para la encriptación de las contraseñas de los usuarios, de esta forma las cuentas serán seguras. Las contraseñas no podrán ser vistas en la base de datos ni siquiera por el propio administrador del sistema.

☐	13	es.support@iriworldwide.com	fc166ea878b2e10bdcd0090e04617e1d	SP
*	(Auto)	*(NULL)*	*(NULL)*	*(NULL)*

Ilustración 20. Ejemplo encriptación en BD

4.4.4. Envío de correos electrónicos

Por último, cabe destacar el uso de correos electrónicos para la notificación del estado de sus peticiones a los usuarios de la plataforma. Para ello se utiliza, como se detalla en el apartado de arquitectura del sistema, un servidor SMTP. En la siguiente imagen se puede ver un ejemplo de como se ha implementado el envío de mails desde el sistema.

```
ini_set('SMTP','localhost');
ini_set('sendmail_from', 'no-reply@iriworldwide.com');

$to = $username;
$subject = 'We changed the status of your Riss with number: ' .$numriss;
$body = "Please check on this website the status of your RISS: http://spintranet/apps/riss/index.html" ;
if(!mail($to, $subject , $body))
    {
        echo "Error to send the mail";
    }
```

Ilustración 21. Ejemplo de envío de mail.

4.5. Validación / pruebas

El *software* RISS es una aplicación *web* de petición de *software* con accesos de usuarios y editor y distintas funcionalidades para cada uno de ellos.

Como apoyo a su funcionamiento se utiliza una base de datos externa en un servidor Windows 2012 donde se guardan los datos necesarios.

Se realizarán pruebas para las tres partes diferenciadas, usuarios y editores *helpdesk* y financieros. Entre las diferentes pruebas estarán el inicio de sesión, la correcta edición de peticiones, y cumplimiento de requisitos como la longitud del nombre de usuario, contraseña, preguntas o respuestas.

El usuario podrá seleccionar un *software* que no esté en la lista de predeterminados, no se comprobará su existencia de forma desatendida.

Robustez, portabilidad y rendimiento de la aplicación y la base de datos, pérdidas de conexión o respuestas ante cierres inesperados serán los principales pilares de la batería de pruebas propuestas.

Se han realizado tres tipos de pruebas generales. Pruebas de validación que comprueban la actuación del *software*. Pruebas de verificación en las que demostraremos que cumple con los requisitos definidos por la empresa cliente. Prueba de fallos básicas para el correcto funcionamiento del *software*.

4.5.1. Restricciones

Las pruebas unitarias se han realizado tanto con el programa en local como en el propio *software* montado en el servidor de la empresa cliente, por lo que la restricciones serán las propias del *software* utilizado. Se ha seguido una metodología **Bottom – up**, comprobando

primero componente a componente para subir a funciones y finalmente llegar a probar el programa completo [9].

4.5.2. *Software* a probar

En la siguiente tabla se resumen las pruebas a realizar. Se agrupan por componente.

Componente	Prueba
Menú	Acceso al menú correcto dependiendo del tipo de usuario
Usuario	Pruebas varias (añadir petición, ver, ver selección…)
Inicio de sesión	Entrada al sistema con datos correctos
Contraseña	Cambio de contraseña y recordatorio por el correo electrónico.
Registro	Registro de usuario *helpdesk* (Solo el administrador tiene permisos de creación)
Usuario *helpdesk*	Pruebas varias (modificar petición, ver, ver selección del propio país del user…)
Abandono	Cerrar sesión
Usuario financiero	Pruebas varias (modificar petición, ver, ver selección…)
Excel	Exportación Excel en formato .xslx
Test	Correcto funcionamiento del flujo de la petición
Estadísticas	Acceso a gráficas y correcto modificación
Vista	Nombre de menús

Tabla 12. Componentes a probar

4.5.3. Pruebas validación

Como se menciona en este mismo apartado, se entiende como pruebas de validación las que comprueban la actuación del *software*; es decir, se comprueba que el *software* cumple las expectativas de la empresa.

Desde la empresa cliente se detalla una lista de pruebas a realizar, en este apartado solo se detallan las más relevantes para el sistema.

Identificador	RISS-PV-01
Nombre	Prueba de validez al añadir petición por usuario.
Entrada	Nombre de usuario con el siguiente formato:

	nombre.apellido@dominio.com Se comprobará el punto y el "@". Envío de un correo electrónico al usuario final.
Prueba	Comprobar que la petición puede ser creada con ese nombre de usuario.
Salida	True si es correcta, false si es erróneo y retroceder para volver a intentarlo sin perder el resto de datos del formulario.

Tabla 13. RISS-PV-01

Identificador	RISS-PV-02
Nombre	Prueba de validez petición por usuario
Entrada	Nombre de departamento, nombre del manager de departamento, selección correcta del país. Mínimo un *software* por petición, cantidad de *software* requerido, uso temporal y fecha hasta la utilización o no. Justificación obligatoria.
Prueba	Comprobar que la petición puede ser creada con todos los datos necesarios.
Salida	True si es correcta y envío de correo electrónico al usuario que envía la petición, false si es erróneo y retroceder para volver a intentarlo sin perder el resto de datos del formulario.

Tabla 14. RISS-PV-02

Identificador	RISS-PV-03
Nombre	Prueba de validez inicio de sesión.
Entrada	Nombre de usuario *helpdesk* correcto creado previamente por usuario administrador.
Prueba	Visualizar todas las vista de *helpdesk* y todas sus funciones (por país al que pertenece)
Salida	True si es correcta, false si es erróneo o no existe usuario creado.

Tabla 15.RISS-PV-03

Identificador	RISS-PV-04
Nombre	Prueba de validez cambiar contraseña.
Entrada	El usuario *helpdesk* accede a la opción de cambio de contraseña con las restricciones dadas. Mínimo 8 caracteres sin caracteres " o "–"
Prueba	Inicio de sesión del usuario con su nueva contraseña
Salida	True si es correcta, false si es erróneo y retroceder para volver a intentarlo sin perder el resto de datos del formulario.

Tabla 16.RISS-PV-04

Identificador	RISS-PV-05
Nombre	Interfaz *web*: menú usuarios
Entrada	--
Prueba	Comprobación de que el menú mostrado es el correcto si el sujeto es usuario o editor.
Salida	True si es correcta y, false si es erróneo.

Tabla 17. RISS-PV-05

Identificador	RISS-PV-06
Nombre	Prueba validez registro
Entrada	Solo el usuario *helpdesk* administrador puede crear nuevos usuarios hellpdesk.
Prueba	Inicio de sesión como administrador. Añadir un nuevo usuario hellpdesk con su país asociado.
Salida	True si es correcta y, false si es erróneo o ya existe ese nombre de usuario.

Tabla 18.RISS-PV-06

4.5.4. Pruebas de verificación

Las pruebas de verificación son las que comprueban que se cumplen los requisitos funcionales y no funcionales especificados por la empresa cliente.

Identificador	RISS-PVR-01
Nombre	Comprobación inicio de sesión.
Descripción de la prueba	Comprobación de que el usuario *helpdesk* puede entrar al sistema con sus credenciales.
Componentes involucrados	BD inicio de sesión, interfaz *web*.
Resultado esperado	Usuario y contraseña correctos: Acceso al menú de editor. Usuario o contraseña incorrectos: Mensaje de error y botón de retroceso a la página anterior.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RF-RISS-16.

Tabla 19.RISS-PVR-01

Identificador	RISS-PVR-02
Nombre	Comprobación registro.
Descripción de la prueba	Si el nombre de usuario no coincide con uno existente o la base de datos no está llena, la cuenta se creará correctamente. En caso contrario aparecerá un mensaje de error.
Componentes involucrados	BD inicio de sesión, interfaz *web*.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RF-RISS-20, RF-RISS-21.

Tabla 20.RISS-PVR-02

Identificador	RISS-PVR-03
Nombre	Comprobación test.
Descripción de la prueba	Comprobación de que el usuario pueda enviar una petición sea recibida por el *helpdesk*, modificada y enviada al usuario financiero para su aprobación.
Componentes	BD inicio de sesión, BD ITO, BD swname-table, BD request, interfaz

involucrados	*web*.
Resultado esperado	Realización correcta del test de la petición con su flujo correcto.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RNF-RISS-09, RNF-RISS-10, RNF-RISS-11, RNF-RISS-13, RF-RISS-01.

Tabla 21.RISS-PVR-03

Identificador	RISS-PVR-04
Nombre	Comprobación visualización de las peticiones del usuario.
Descripción de la prueba	Comprobación de que el usuario pueda ver todas sus peticiones independientemente del estado. No podrá modificarlas.
Componentes involucrados	BD swname-table, BD request, interfaz *web*.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RF-RISS-02.

Tabla 22.RISS-PVR-04

Identificador	RISS-PVR-05
Nombre	Comprobación modificación de la petición por parte del *helpdesk*.
Descripción de la prueba	El usuario *helpdesk* podrá modificar y guardar los cambios en la petición. Podrá adjuntar un archivo. Siempre será propia del usuario de cada país.
Componentes involucrados	BD inicio de sesión, BD ITO, BD swname-table, BD request, interfaz web.
Requisitos validados	*RFN-RISS-14 , RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29,* RNF-RISS-05, RNF-RISS-11, RNF-RISS-13, RF-RISS-06, RF-RISS-07, RF-RISS-08, RF-RISS-09, RF-RISS-10, RF-RISS-11, RF-RISS-12, RF-RISS-13, RF-RISS-14.

Tabla 23.RISS-PVR-05

Identificador	RISS-PVR-06
Nombre	Comprobación correo electrónico.
Descripción de la prueba	Se deberá activar la opción de protocolo SMTP en el servidor para poder realizar el envío de correos electrónicos a cada usuario.
Componentes involucrados	Gestor de correo, interfaz *web*.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RF-RISS-03.

Tabla 24.RISS-PVR-06

Identificador	RISS-PVR-07
Nombre	Comprobación manual.
Descripción de la prueba	Se podrá consultar el manual de usuario en cualquier momento de la aplicación.
Componentes involucrados	Interfaz *web*.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RNF-RISS-07, RNF-RISS-08.

Tabla 25.RISS-PVR-07

Identificador	RISS-PVR-08
Nombre	Comprobación de estadísticas.
Descripción de la prueba	Se podrá consultar las estadísticas, siempre actualizadas tanto en la vista de *helpdesk* como en la del usuario financiero.
Componentes involucrados	BD inicio de sesión, BD ITO, BD swname-table, BD request, interfaz *web*.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RF-RISS-15, RF-RISS-22, RF-RISS-23.

Tabla 26.RISS-PVR-08

Identificador	RISS-PVR-09
Nombre	Comprobación de exportaciones.
Descripción de la prueba	Se podrá exportar los datos de la base de datos por parte del usuario financiero.
Componentes involucrados	BD inicio de sesión, BD ITO, BD swname-table, BD request, interfaz *web* y plataforma que permita abrir hojas de cálculo.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RF-RISS-28.

Tabla 27.RISS-PVR-09

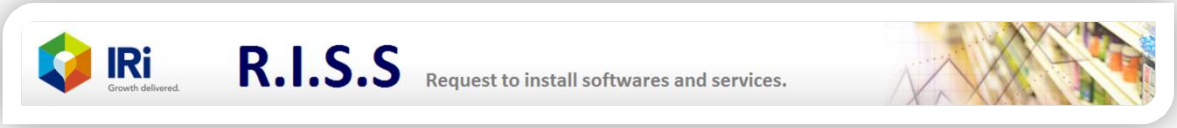
Identificador	RISS-PVR-10
Nombre	Comprobación de logout.
Descripción de la prueba	Se podrá hacer logout en cualquier momento.
Componentes involucrados	BD inicio de sesión, interfaz *web*.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RF-RISS-19.

Tabla 28.RISS-PVR-10

Identificador	RISS-PVR-11
Nombre	Comprobación de interfaz.
Descripción de la prueba	Se probará que todos los botones de la interfaz *web* cumplen su función.
Componentes involucrados	Interfaz *web*.
Requisitos validados	RFN-RISS-14, RFN-RISS-15, RFN-RISS-16, RFN-RISS-17, RF-RISS-03, RF-RISS-29, RNF-RISS-06.

Tabla 29.RISS-PVR-11

Capítulo 5: Conclusiones y trabajos futuros



5.1. Conclusiones

Este proyecto se ha realizado satisfactoriamente y en el tiempo especificado por la empresa cliente. Se cumplen todos los requisitos especificados, entre ellos la funcionalidad de la gestión de *software*.

Algunas de las novedades que diferencian a esta plataforma de cualquier otra de pago o *software* libre disponibles en la red son las siguientes:

- la introducción de gráficos para la visualización y análisis de los datos recogidos por los usuarios.
- La petición solicitada pasa por varias fases (descritas en este proyecto), es decir, el flujo seguido es muy específico y no se corresponde con ninguno otro encontrado en cualquier *software* libre o de pago.
- Diferenciación por países y acceso a un único usuario *helpdesk* representante de cada país.
- La exportación de los datos a Excel para su análisis.
- Facilidad de uso para el cliente.

Todas estas novedades hacen de esta plataforma una aplicación necesaria, cumpliendo con el objetivo de la reducción de tiempo para la empresa cliente debido a su requisito fundamental: la reducción de costes.

La aplicación web cumple con el diseño planteado de un modelo cliente/servidor, de este modo todas las llamadas a la base de datos se realizan desde la página, utilizando un servicio web.

5.2. Conclusions

This project was realized satisfactorily and it was in time specified by the company client. All specified requirements, including the functionality of the software management have been met.

Some of the innovations that differentiate the platform from any other payment or free software available are:

- The graphics and viewer of the analysis of the data collected by users.
- The flow of a request goes by some phases (specified on this document). This workflow is very specific and does not correspond with any other software on the net
- The difference between countries and the access to a single helpdesk user of each country.
- The export of the data to Excel for his analysis.
- The ease use for the final client.

All these new features make necessary this application, fulfilling the objective of reducing the time for the client company because of its fundamental requirement: reducing costs

The web application carries out with the raised design of client/server model. All the calls to the data base will be made through the web application, using the web service.

5.3. Trabajos futuros

La plataforma desarrollada, pese a cumplir con los requisitos del cliente, es susceptible de mejorar.

La posibilidad de la realización de nuevos trabajos o modificación de los ya existentes, es viable gracias al modo en el que ha sido diseñada la aplicación. La estructuración y la facilidad de lectura del código hacen posible su entendimiento, requiere un mínimo esfuerzo por parte de programadores integrar nuevas funcionalidades a la plataforma. Entre las posibles mejoras destacan dos:

- Visión compacta y reducida para móviles
- La posibilidad de creación de gráficos propios seleccionando los datos deseados.

Uno de los componentes principales del cambio tecnológico de hoy en día es la utilización de móviles y *tablets* como vía de trabajo convencional, dando movilidad a los usuarios. Este es un punto importante si hablamos de una consultoría como empresa cliente, donde los usuarios son comerciales. Poder revisar tu estado de peticiones o completar otras, desde un dispositivo móvil es, por tanto, un camino interesante donde seguir trabajando. La vista compacta tendría las mismas vías de trabajo que la versión para ordenador pero con una estética más sencilla, facilitando al trabajador la utilización de dicha plataforma.

En la aplicación web actualmente se pueden visualizar tres gráficos especificados por la empresa cliente, entre ellos: el coste por país o el coste por *software* solicitado. Como trabajo futuro se desea que el usuario seleccione cualquier combinación de datos, permitidos para dicho cliente, (por ejemplo: coste y país, coste y usuario final) y crear con ellos diferentes tipos de gráficas para su análisis.

5.4. Marco Regulador

Será de obligatorio cumplimiento la protección de los datos en toda la aplicación, no revelando información de los usuarios ni de los editores. Se cumple así la Ley Orgánica 15/1999, de 13 de diciembre, de Protección de Datos de Carácter Personal.

"La presente Ley Orgánica tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar" [11]

La aplicación web, RISS, cumple la Ley de Protección de Datos de Carácter Personal. Uno de los ejemplos que podemos destacar en este campo es la encriptación de las contraseñas con MD5, lo que supone que las contraseñas no sean reveladas al usuario final en ningún momento. Tampoco será posible su visualización en la base de datos. Además, para la creación del usuario final será necesaria la intervención del administrador, quien crea el usuario.

Esta ley también se aplica al marco de la consultoría, donde los usuarios de dominio tienen una serie de privilegios, aun así, éstos no podrán extraer información de la plataforma para sacarla al exterior por política de la empresa.

5.5. Entorno socioeconómico

En los últimos años la situación socioeconómica se encuentra dentro el contexto de la crisis económica mundial.

Todo el mercado ha sido sacudido por esta crisis, de tal manera que se han modificado las conductas habituales, con la premisa del ahorro de costos organizativo.

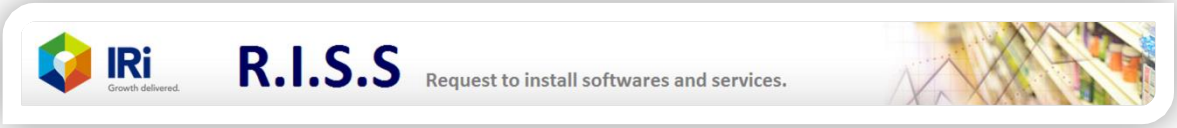
La empresa cliente, consultora de mercado, tiene que competir en un mercado globalizado, lo cual obliga a ajustar los costes de los procesos de administración de sistemas informáticos.

Siendo además una empresa internacional quien encarga el proyecto, todo lo que permita ahorrar tiempo en la comunicación y globalización de los procesos será bienvenido a la hora de ahorrar costes.

El planteamiento de la empresa cliente, para este escenario, se basa en la optimización de recursos, que ahorren tiempo de gestión y hagan posible controlar debidamente los gastos.

El disponer de un sistema que ayude en la gestión de incidencias permite ahorrar costes y reducir la tasa de fallos en dichos procesos. En consecuencia la empresa puede hacer frente a sus competidores.

Capítulo 6: Anexos



6.1. Global abstract

6.1.1. Introduction

We live in the information and computerization society and the most basic tasks is becoming a necessity for the proper development of these. We need the help of the technology and the conventional methods of sending information, so sharing information in paper have become obsolete.

It is customary that the waits are shorter every day. We have left behind, the age when we needed several days of waiting for a single answer. The immediacy transfer of information translates into a cost reduction and resource optimization for companies, which makes this technique more necessary.

Forms validation, web pages or applications for mobile or Tablet are one of the solutions for these problems. Prioritizing the importance of each management and allowing the end user to give a quick response, operators avoiding the need for direct contact with customers.

The project whose code name is "RISS", aims to design, implementation and commissioning a web application for managing software, requested by the company client, the consultancy IRI Worldwide.

This application will aim to support and reinforce, allowing users to accelerate a previously much more costly process, in time and resources. Otherwise, user request and their alterations or approval will be added by users with special privileges called "editors". These editors can value the request and analyzes the statistics through graphics. The number of requests, the total cost per country or the reduction or amplified of the cost can be evaluated with this method.

The project began with the assignment of resources by the client company, after which the planning it was made. Then, defining the requirements of the application, the design of it, it is implemented in individual blocks for the subsequent integration of the complete system. After completing the integration with the operating system, it will be tested with a battery of modular tests designed previously. Finally, the software supplied with the relevant documentation for further evaluation by the customer.

The part of the client company assumes that team members are familiar with application development programming web pages, as well as the efficient management of time for tasks completion before the deadlines set. Similarly, it is known that members of the team carry a load outside the project would consume much of your time and can be limiting for

the same activities; however, it is expected that this limitation is not the cause for delay in corresponding deliveries or improvement in software quality.

6.1.2. Objectives

The objective of this project is developing web application management software to fully satisfy the demands required by the client company. The creation of "RISS" software in a parallel way intended to demonstrate the knowledge acquired in the technological area as work that composes in the management and development of a big size software project, in order to be prepared for incorporation into professional software world once finished the period of instruction.

The basic function of the web application can be summarized in:

- Control and maintenance of project management software and licenses of the company.
- Allocating staff, costs registration and time dedicated to specific project.
- Management costs charged to the projects.
- Analysis by date and country of costs incurred.
- Possibility of exporting data to Excel invoice mode.

The platform need to be safety, fulfill the privacy and security requirements of end users, without disclosing information and have no breaches in data protection or passwords.

Definitely, the main objective of the application is to reduce the time of the company staff involved.

In a parallel way, the objectives of the project and final degree work are summarized in:

- Understand what is a web environment and how it works
- Learning implementation techniques with PHP, HTML, CSS, JavaScript and MySQL
- Management, modification and processing of data.
- Design and structuring of a big size project for a client, a consulting company IRI worldwide.

6.1.3. A new platform is viable

This project focuses on the framework of a consulting firm IRI Worldwide. IRI is a business consulting firm that dedicates its studies to the sectors of food and drugstore. They include world renowned clients such as Coca-Cola, Mercadona or Schweppes among others.

In recent years there has been a shift in the economy of 180degrees. Given this new reality where consumers change their buying behavior, manufacturers and distributors cannot wait weeks or months to react. To respond quickly, market consultants, as IRI Worldwide, which allows its clients to go one step further of its competitors.

This is an American company. It has two offices in Spain, which work in tandem on common projects. They are located in Madrid and Barcelona. Their communication is essential for the proper development of studies and works. Communication programs are the key to your day. This company carries out projects on an international scale, with France, Italy or India, where the headquarters of technical department and support is placed.

The operation of the company can be summarized in a few lines. The companies expect to receive information from point of sale, and then analyze it. After examination, within a few weeks the results will be invalid. The main role of IRI is to facilitate this information as quickly as possible before it loses value.

Definitely, the key to success of a consulting is the reaction time, which transforms in reduction of costs. The role of a technical in a company like this is fundamental, because will be who can transform communication in an immediate process.

This project focuses on communication between several departments to control software, licenses and therefore costs. Although before deciding to create a new platform, the existing products which are already on the market, have been studied.

Before considering the possibility of creating a new platform, we have studied other programs with similar objectives, existing in the market, like: OSTICKET[3], GLPI[4], SPICEWORKS HELLP DESK[5], REDMINE[6]…

All of them are quality software but the common problem is that do not reunite the basic requirements of the client company. First, all these platforms are software for managing incidents and not for managing software and therefore the application flow does not conform to the required.

There is no division by departments, it is not found the possibility of extracting data invoice mode, controlling costs through statistics, or there is no real reduction in time with the use of these applications, either by the amount of emails sending or difficulty who show to the end user. For all these reasons, the RISS project needs a new platform designed and built specifically to this company.

6.1.4. RISS platform, the characteristics

It is used a cascading model. This model is the development of software for different and separate stages. Previously, the specification of requirements was needed by the client company. Each project phase should wait for completion of the previous one.

The client company is responsible for carefully detail all the requirements that your system must have. After evaluation and discarding some requirements, we proceed to his definition.

Summarize the requirements of the division into three users and therefore three different views of the application. End user, user helpdesk and user finance department.

The stage of software and system designs, the system and component parts is organized, defining the structure of the solution and implement the chosen architecture.

Through the chosen pattern MVC or model-view-controller, it will be established a design that reunites these characteristics.

The RISS platform was made with specific language of programing required by the client. Firstly, the client part was made with HTML, CSS and JavaScript language. This part of the architecture concurred with the view.

The backend of the platform was built with PHP modules. Note that in the customer requirements exist a specific point for downloading the data and exporting them to Excel (RF-RISS-28). For this, the module used PHPExcel.

The software needs a connection with a database hosted on an external server. Installation of the database and the project was carried out on a Windows Server 2012 [14], server is determined by the client company. The design and management of application data were made through a MySQL database. This data base was formed with a set of tables, with the main feature of following the entity-relationship model.

This document details how the application is designed safely. Users privacy is an important point in the whole platform. Using this application cannot be a source of loopholes in enterprise security.

For user passwords it used the MD5 encryption method, in this way accounts are used safe. Passwords cannot be seen in the database not even by the administrator of the own system.

To solve the requirement for generation of graphics (RF-RISS-15) is used Highcharts [8] API. It works on all desktop or mobile browsers, like iPhone or iPad. The license is free and only need two JavaScript files to run: the highcharts.js and either core jQuery, MooTools or Prototype framework. To use this API must only include the JavaScript library and define a new instance Higthcharts.

Finally, we note the use of email for notification of the status of their requests users of the platform. For this purpose an SMTP server is used.

6.1.5. Battery of tests

The RISS software is a web application for software request with user and editor access. For testing the reliability of the software, a set of tests will be performed.

Tests for three different parts, helpdesk and financial users and publishers will be made. Among the different tests will login, the correct edition of requests, and compliance of requirements such as the length of the username, password, questions or answers.

Robustness, portability and performance of the application and the database lost connections or responses to unexpected closures are the main pillars of the battery of tests proposed.

There have been three types of general tests. Validation tests that checks the performance of the software. Verification tests demonstrate that it reunites the requirements defined by the client company. Basic tests failures for the proper functioning of the software.

This project was realized satisfactorily and it was in time specified by the company client. All specified requirements, including the functionality of the software management have been achieved.

6.2. Anexo 2: Tabla de precios

	Concepto	Dato de partida					coste empresa
Recursos Materiales	PC Dell Optiplex 740	Precio de mercado					700
	PC Servidor Windows Server 2012	Precio de mercado					700
	Teclado	Precio de mercado					35
	Ratón	Precio de mercado					15
			bruto anual	hr anuales	precio/hora	K	coste empresa/h
	Hr. Director del Proyecto	(*) Nivel 2. Diplomados y titulados 1.er ciclo universitario. Jefe Superior.	*17.544,24 €*	*1800*	*9,75 €*	*1,30*	*12,67 €*
Recursos Humanos	Analista	(*) Nivel 1. Licenciados y titulados 2.º y 3.er ciclo universitario y Analista.	*17.544,24 €*	*1800*	*9,75 €*	*1,30*	*12,67 €*
	Programador	(*) Nivel 1. Licenciados y titulados 2.º y 3.er ciclo universitario y Analista.	17.544,24 €	*1800*	*9,75 €*	*1,30*	*12,67 €*
	Diseñador	(*) Nivel 1. Licenciados y titulados 2.º y 3.er ciclo universitario y Analista.	17.544,24 €	*1800*	*9,75 €*	*1,30*	*12,67 €*
	Ingeniero Senior Tutor UC3M	(*) Nivel 2. Diplomados y titulados 1.er ciclo universitario. Jefe Superior.	*23618,28*	*1800*	*13,12 €*	*1,30*	*17,06 €*
	Ingeniero Senior Tutor IRI WORLD WIDE	(*) Nivel 2. Diplomados y titulados 1.er ciclo universitario. Jefe Superior.	*23618,28*	*1800*	*13,12 €*	*1,30*	*17,06 €*
	Director Recursos Humanos Empresa cliente	(*) Nivel 2. Diplomados y titulados 1.er ciclo universitario. Jefe Superior.	*23618,28*	*1800*	*13,12 €*	*1,30*	*17,06 €*

				coste empresa
	Licencia de Windows 2007 para el PC de trabajo	Precio de mercado		174,17 €
Gastos de licencia	Licencia de Windows server 2012	Precio de mercado		259,74 €
	Notepad++	Precio de mercado		- €
	Internet Explorer	Precio de mercado		- €
	Opera	Precio de mercado		- €
	Chrome	Precio de mercado		- €
	Sqlyog	Precio de mercado		- €
	Licencia PHP, HTML, JavaScript, PHPExcel	Precio de mercado		- €
	Project professional 2010	Precio de mercado		1.369,00 €
	Office proffesional plus 2010	Precio de mercado		456,66 €

(*) Según XVII CONVENIO COLECTIVO NACIONAL DE EMPRESAS DE INGENIERÍA Y OFICINAS DE ESTUDIOS TÉCNICOS

Tabla 30. Tabla de precios

6.3. Anexo 3: Manual de usuario

En este documento se explica detalladamente el uso de la aplicación *web* de gestión de *Software*, RISS.

Ya que esta aplicación ha sido diseñada para un uso internacional, se establece como requisito que todo su contenido sea en inglés.

En todas las vistas posibles de la plataforma hay un acceso directo a guía de usuario.

Se va a dividir el manual en tres partes diferenciadas, según sus posibles usuarios:

Lea atentamente este documento con el fin de garantizar el uso correcto y seguro de la plataforma.

Usuario

En primer lugar, se encontrará con una página como la que puede encontrar en la ilustración de abajo. Debe seleccionar una de las tres opciones posibles: nueva petición, listar todas sus peticiones o guía de usuario.

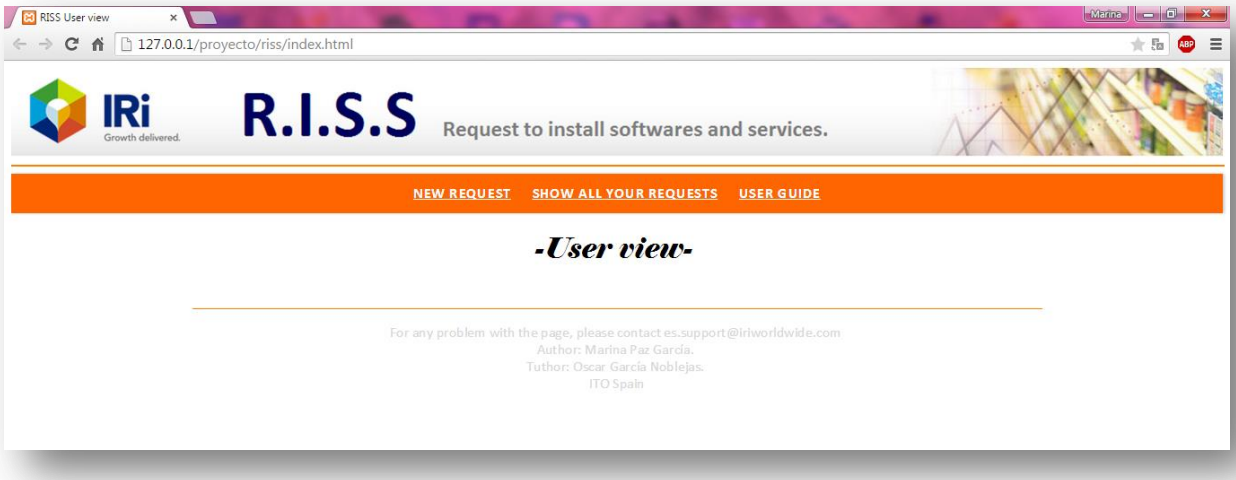


Ilustración 22. Vista principal usuario

Si selecciona nueva petición, encontrará un formulario que debe rellenar. La propia página le indicará que campos son obligatorios, no permitiéndole guardar la petición sin esa información. Deberá completar el nombre de usuario como se indica en el ejemplo, con su usuario de correo de la empresa. Si el correo no es correcto la petición quedará invalidada.

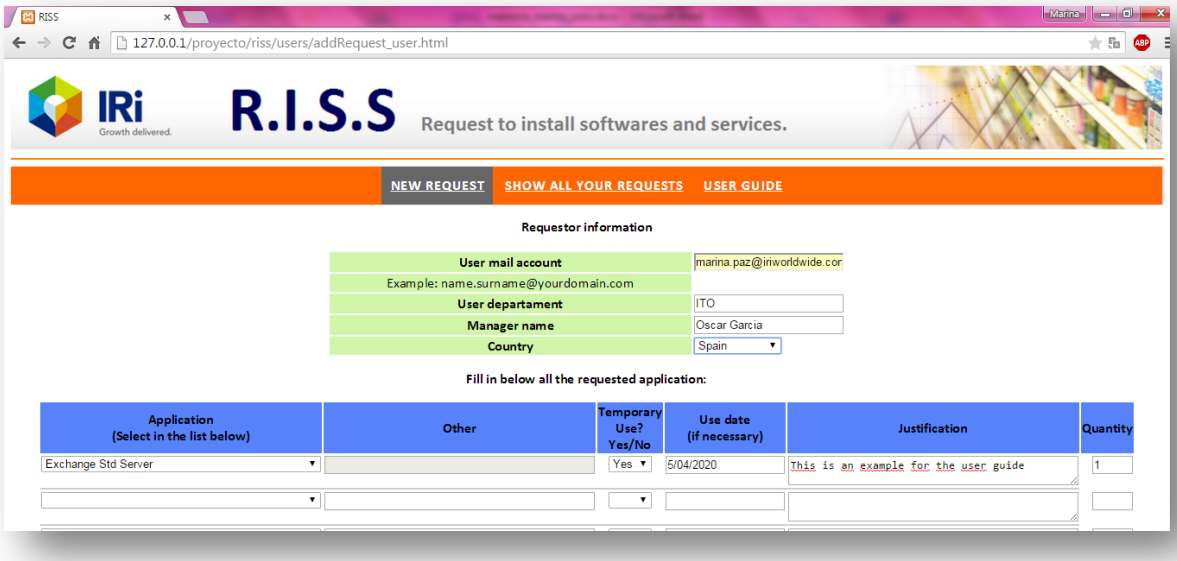


Ilustración 23. Nueva petición

Si la petición es guardada con éxito y no hubo ningún problema recibirá un correo informando con su número de petición para poder buscar y consultar su estado más adelante.

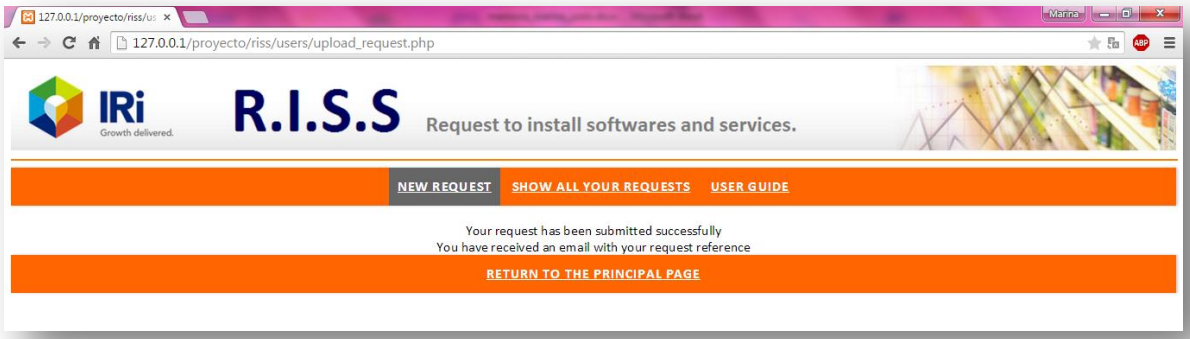


Ilustración 24. Nueva petición guardada

Para listar las peticiones de usuario, pinche en "show all your request".

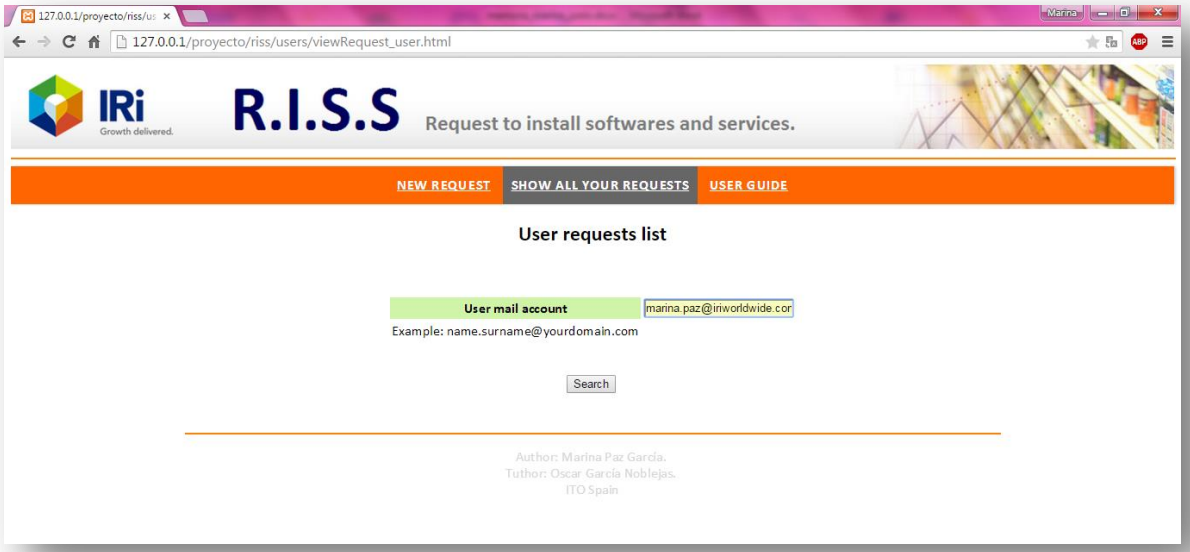


Ilustración 25. Mostrar todas tus peticiones

Encontrará una pantalla como la de arriba, donde deberá introducir su nombre de usuario con el que guardó la petición.

Encontrará una lista con todas las peticiones solicitadas. En el correo que la aplicación le envío una vez guardada la solicitud se muestra el código de su petición de la forma: "RISS+ INICIALES DE SU PAÍS + ID ", únicamente debe recordar ese número para identificar la petición buscada.

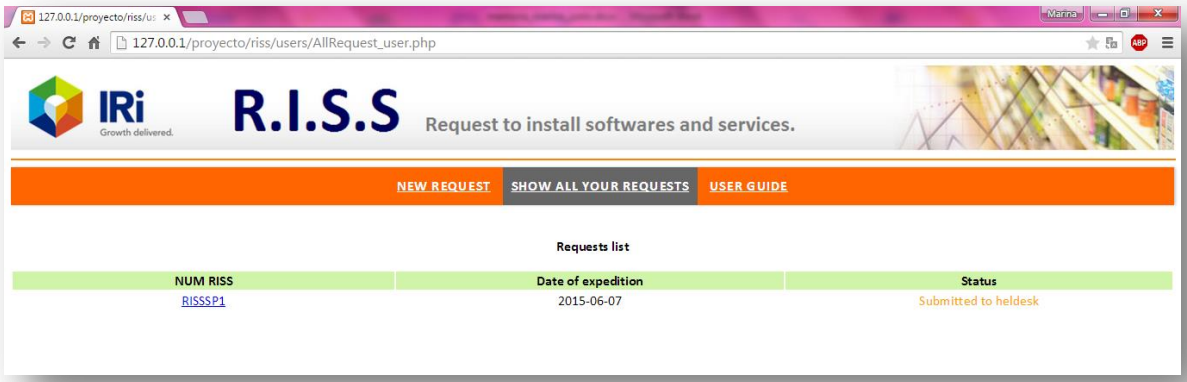


Ilustración 26. Lista peticiones/usuario

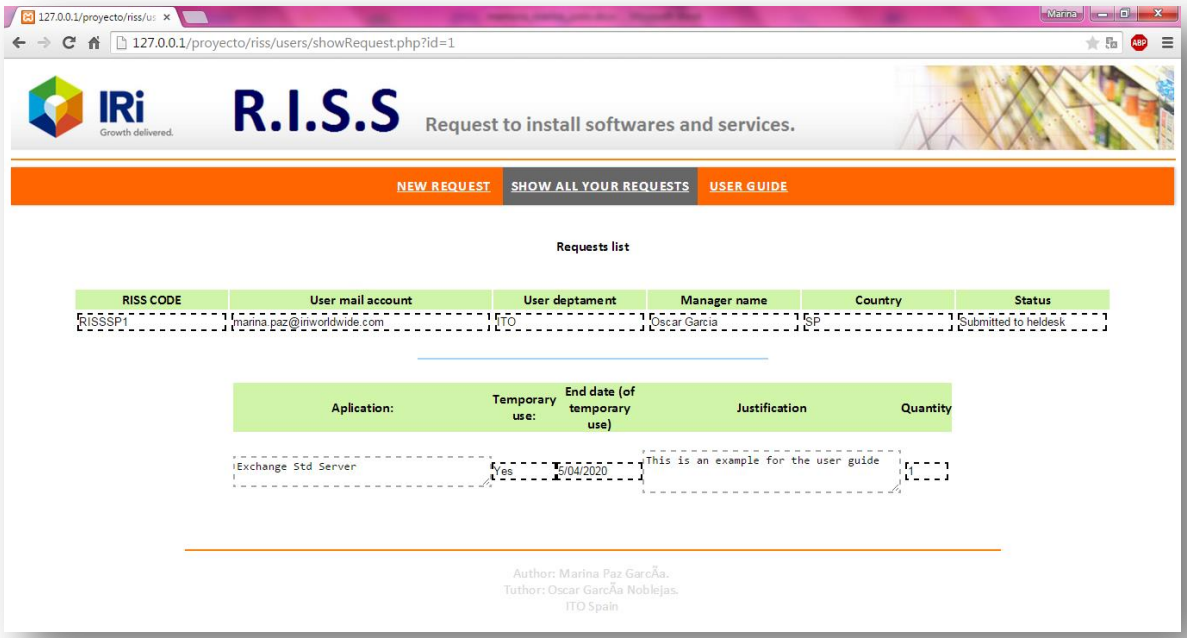


Ilustración 27. Resumen de petición

Helpdesk

El usuario *Helpdesk* se encontrará con una página de inicio como la que se adjunta en la siguiente imagen. Deberá tener un usuario y contraseña el cual previamente debe solicitar al usuario administrador del sistema. Solo existirá un único usuario *helpdesk* por país y este solo podrá ver y/o modificar información de su país.

A continuación se enumeran detallan las opciones más relevantes.

HLP.1. Iniciar sesión

HLP.2. Si es el administrador, registrar un nuevo usuario.

HLP.3. Enviar un recordatorio de la contraseña vía mail

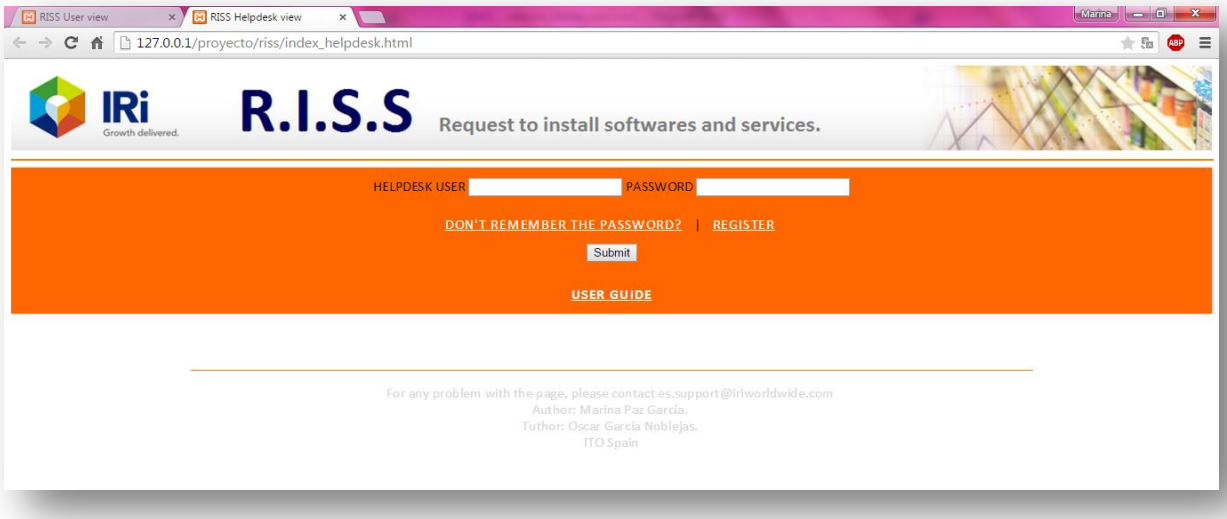


Ilustración 28. Vista helpdesk

Una vez dentro del sistema con el usuario *logeado* podrá navegar en una pantalla como esta:

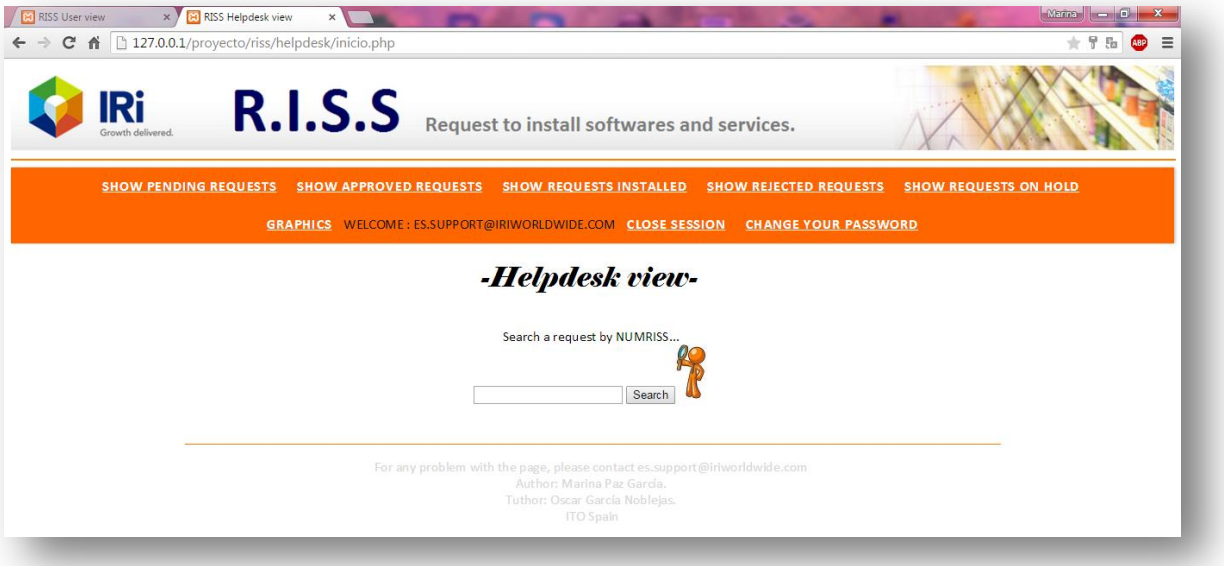


Ilustración 29. Helpdesk logeado

El usuario podrá.

HLP.4. Ver las peticiones pendientes

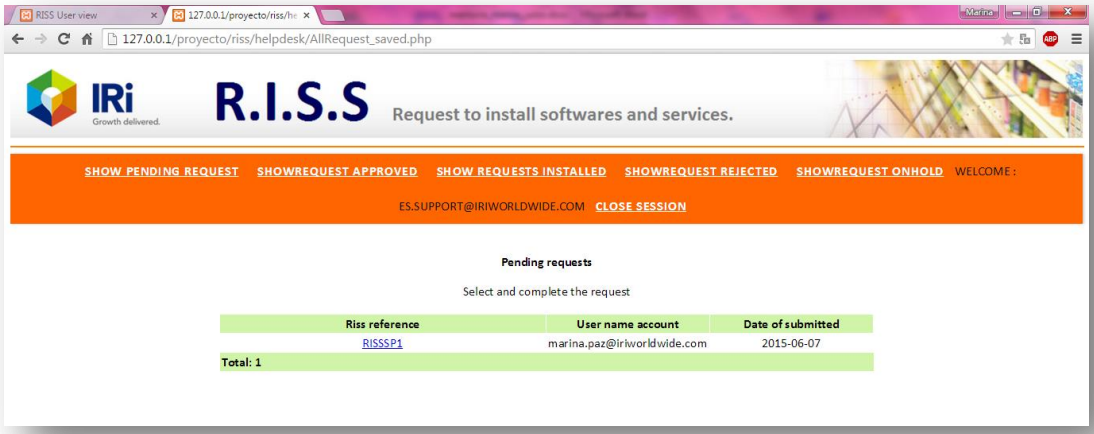


Ilustración 30. Peticiones pendientes

HLP.5. Modificar las peticiones y guardarlas

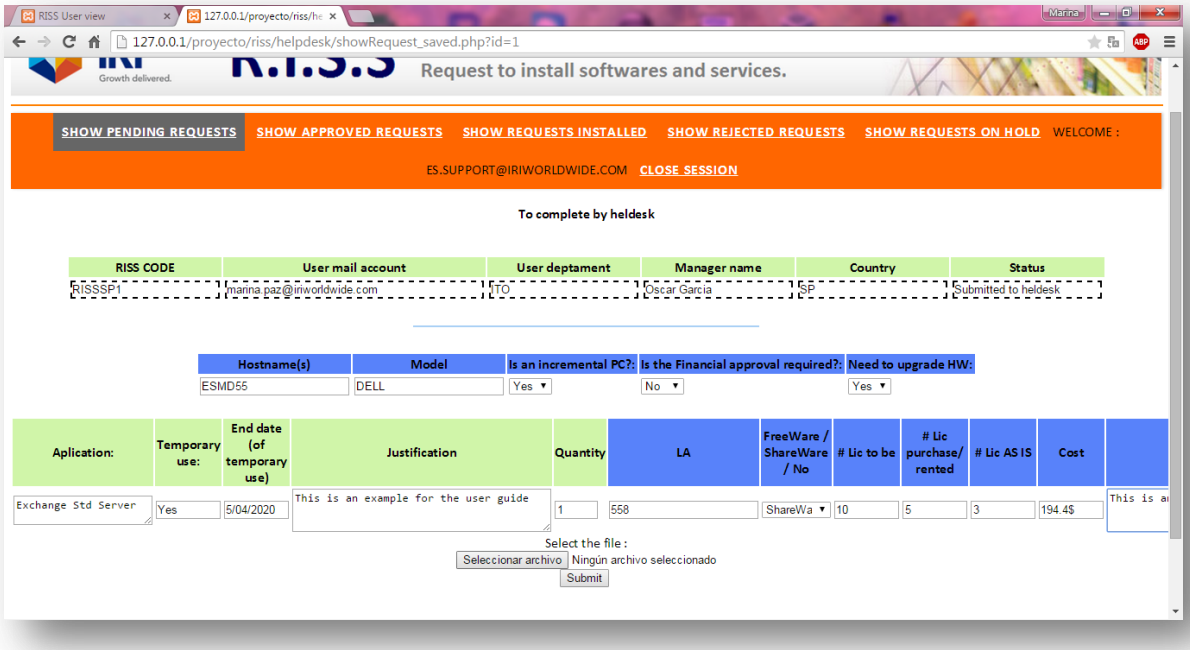


Ilustración 31. Modificar petición y guardarla

Además podrá completar la petición subiendo un archivo de texto, PDF, Excel…

Nota: El único archivo que no será leído por el navegador, aunque si puede ser subido al servidor y abierto con Outlook, es el de .msg

HLP.6. Ver las peticiones aprobadas por financiero

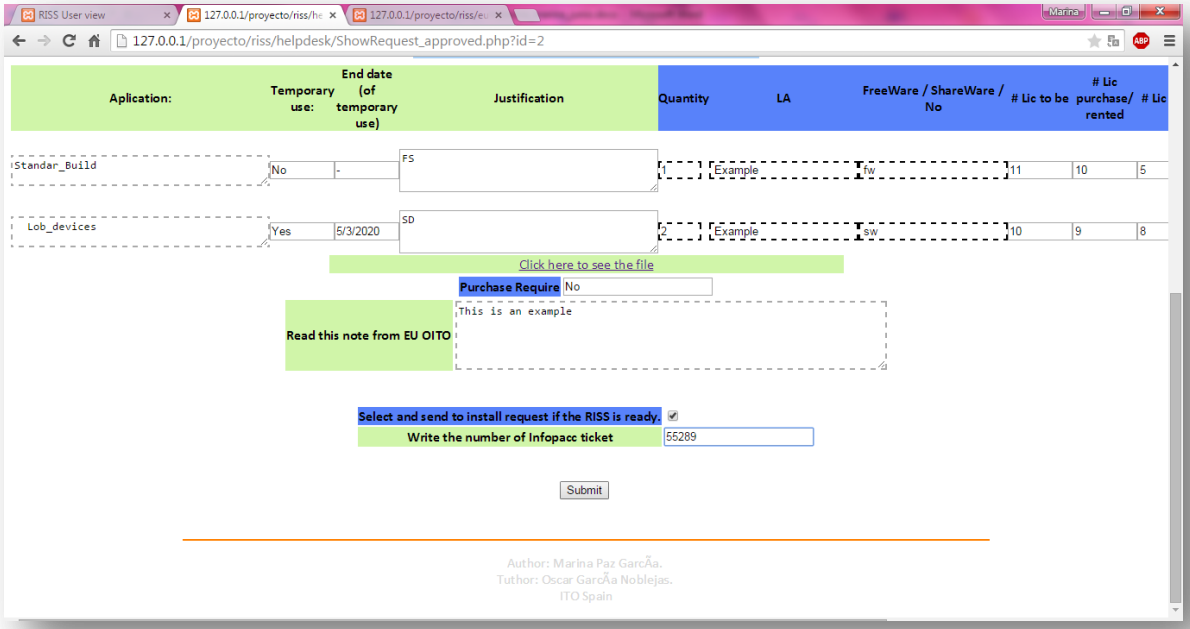


Ilustración 32.Peticiones aprobadas por financiero

Donde podrá seleccionar si desea instalarlo ahora. Para ello seleccione la casilla de instalados e introduzca en el campo de ticket Infopack el número de ticket si es necesario.

HLP.7. Ver las instaladas

HLP.8. Ver las rechazadas

Estas dos últimas no tendrán opción de modificación. Solo podrá visualizar la lista y seleccionar como en el resto de los casos una petición para verla.

HLP.9. Consultar los gráficos de estadísticas

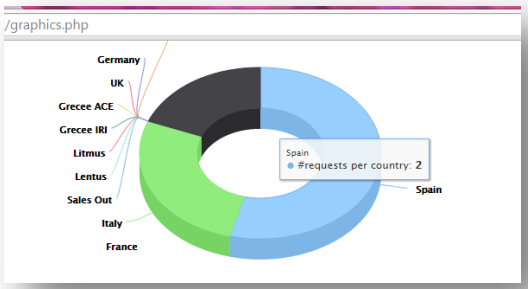


Ilustración 33. Gráfico 1

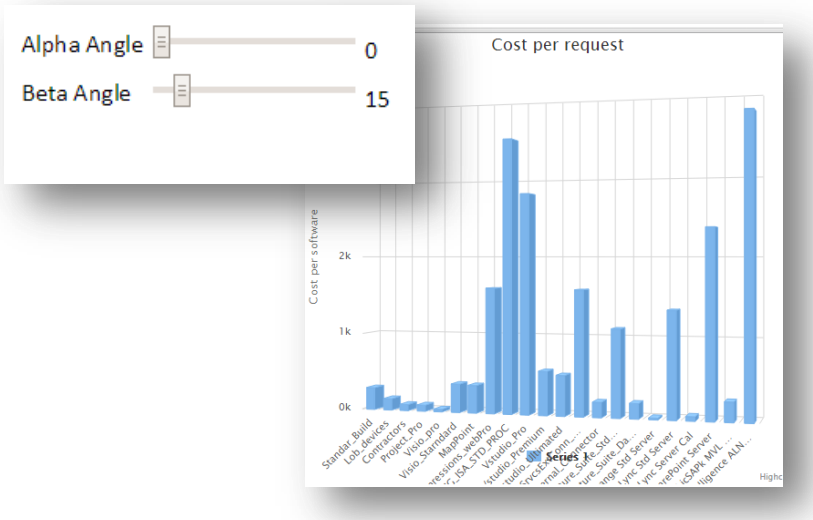


Ilustración 34. Gráfico 2

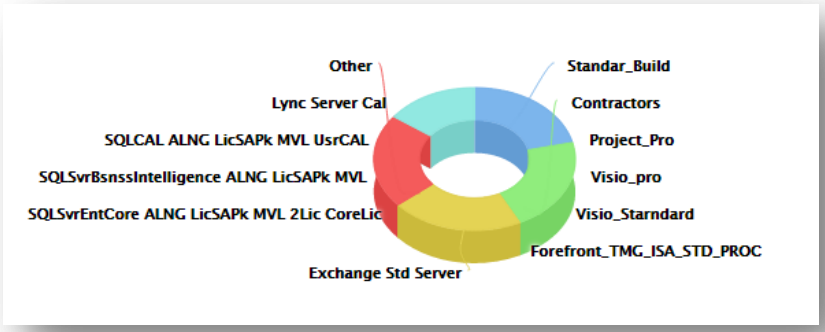


Ilustración 35. Gráfico 3

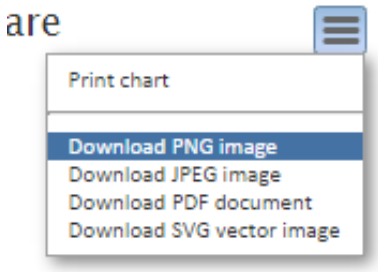


Ilustración 36. Guardar gráficos

Como se observa en la última imagen, podrá guardar en modo PNG, JPEG, PDF o SVG los gráficos.

HLP.10. Modificar las peticiones en espera

En esta ventana podrá modificar todos los campos, desde el campo de notas del departamento financiero podrá observar cuales son los puntos no aceptables en la petición.

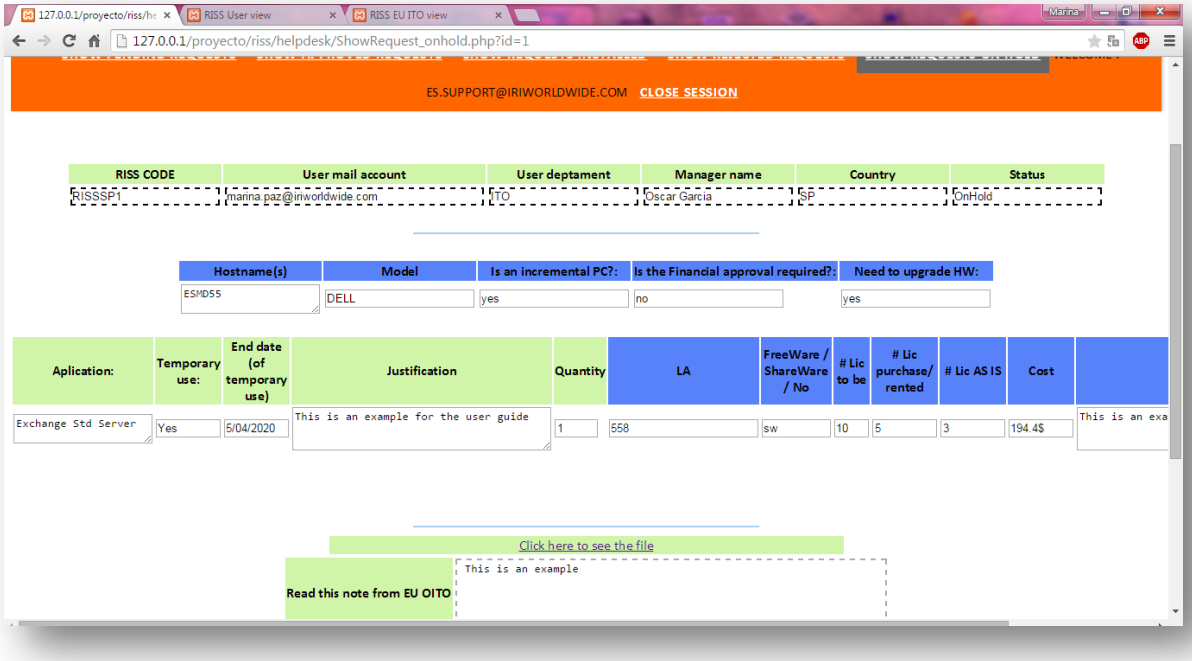


Ilustración 37. On hold

HLP.11. Cambiar su contraseña.

Financiero

El usuario financiero será el encargado de gestionar las peticiones, sus vistas son iguales a las del usuario helpdesk, que puede consultar arriba, como por ejemplo:

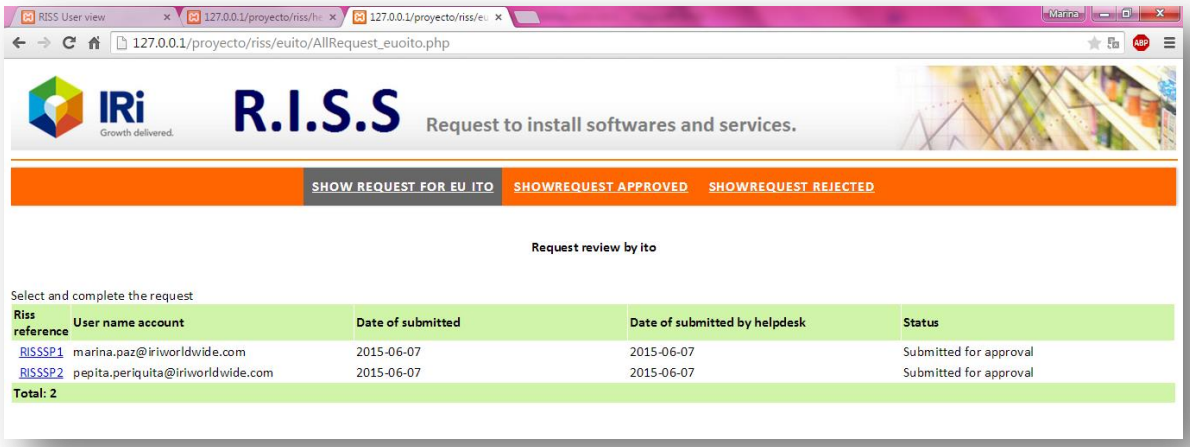


Tabla 31. Peticiones financiero

Su pantalla cambiará a la hora de modificar la petición, deberá examinarla, comprobar los presupuestos y podrá aceptarla, rechazarla o ponerla a la espera.

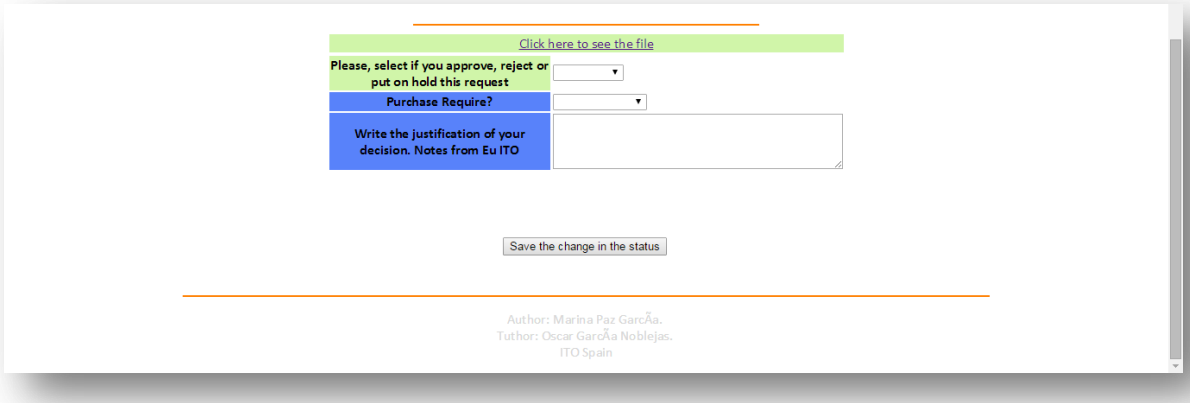


Tabla 32. Decisión de financiero

Dentro de las características especiales del usuario financiero y que distinguen su vista del usuario helpdesk es la posibilidad de exportación de gráficos. Únicamente deberá seleccionar la opción "export" y se le descargara automáticamente un archivo con el contenido actual de las bases de datos (solo se adjunta información relevante para una factura, se obvian contraseñas o información privada).

Lista de acrónimos

Infopacc: Sistema de gestión de incidencias utilizado en la consultoría IRI Worldwide

UML: *Unified Modeling Language*, es un estándar en lenguaje general de modelado orientado a objetos en ingeniería de *software*.

Instancia: Creación a partir de una determinada clase o modelo de un objeto.

MySQL: Sistema de gestión de bases de dato relacional, multihilo y multiusuario bajo licencia GNU GPL.

Relacional: Modelo de base de datos bajado en el uso de relaciones entre conjuntos de datos.

SQL: *Structured Query Language o Lenguaje de Consulta Estructurado*, es un lenguaje declarativo de acceso a bases de datos relacionales desarrollado por IBM.

Swing: es una biblioteca gráfica para Java. Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, desplegables y tablas.

Referencias bibliográficas

[1] Mateu Carles, *Desarrollo de aplicaciones web*, Software Libre XP04/90791/00021,

[2] Ethan Marcotte, "*Responsive Web Design*". A List Appart, issue 306. May 2010. http://alistapart.com/article/responsive-web-design

[3] Peter Lubbers, Brian Albers, and Frank Salim, *Pro HTML Programing, Powerful APIs for Richer Internet Application Development*.

[4] Robert Schifreen. *How to create Web sites and applications with HTML, CSS, JavaScript, PHP and MySQL.* The Web Book.

[5] Tanenbaum Andrew S and Wetherall David J, *Redes de computadores- 5th ed* 2012, México. Pearson, ISBN: 978-607-32-0817-8.

[6] Kurose James F, Keith W. Ross. *Computing Networking: a top-down approach – 5th ed.* 2010, Pearson *ISBN: 0-13-* 607967-9

[7] Gauchat Juan Diego. *El gran libro de HTML5, CSS3 Y JavaScript* - 1º ed 2012. Barcelona. Marcombo S.A. ISBN: 978-84-267-1782-5

[8] Highsoft AS. The Highcharts JS software product. http://www.highcharts.com/products/highcharts

[9] Ian Sommerville y Addison-Wesley. *Software Engineering* -9º ed 2010. Massachusetts, Pearson, ISBN-13: 978-0137035151.

[10] Resolución de 9 de octubre de 2013, de la Dirección General de Empleo, por la que se registra y publica el XVII Convenio colectivo nacional de empresas de ingeniería y oficinas de estudios técnicos. Ministerio de Empleo y Seguridad Social. Boletín Oficial del Estado número 256 (25 de octubre de 2013). Ley del 25 de octubre de 2013 sobre Los salarios pactados en el presente Convenio para el año 2012, en cómputo anual. Boletín Oficial del

Estado. Páginas 86811-86838 http://www.boe.es/boe/dias/2013/10/25/pdfs/BOE-A-2013-11199.pdf

[11] Ley Orgánica 15/1999, de 13 de diciembre, de Protección de datos de carácter Personal. Publicado en BOE núm. 298, de 14/12/1999, Documento consolidado BOE-A-1999-23750. http://www.boe.es/buscar/act.php?id=BOE-A-1999-23750

[12] Bernd Bruegge y Allen H.Dutoit. *Object Oriented Software Engineering - 3º ed 2010*. Pearson, ISBN: 978-0-13-606125-0

[13] Satish Mishra. *Visual Modeling & Unified Modeling Language (UML): Introduction to UML*. Rational Software Corporation. Accessed 9 November 2008.

[14] Universidad de Vigo, Área de Ciencias de la Computación e Inteligencia Artificial *Sistemas Cliente – Servidor, Informática* - http://ccia.ei.uvigo.es/

Enlaces de interés.

1. BestPractical → https://www.bestpractical.com/rt/
2. Otrs → https://www.otrs.com/?lang=es
3. Osticket → http://osticket.com
4. GLPI-Project → http://www.glpi-project.org/spip.php?rubrique18
5. Spiceworks → http://www.spiceworks.com/free-help-desk-*software*/
6. Redmine → http://www.redmine.org
7. Oracle → http://www.oracle.com/us/sun/index.htm
8. Highcharts → http://www.highcharts.com/products/highcharts
9. Mootools → http://mootools.net/
10. Excanvas → http://excanvas.sourceforge.net/
11. Perl → https://www.perl.org/
12. PhpExcel → https://phpexcel.codeplex.com/

13. Http y aplicaciones web, Universidad de Alicante
http://rua.ua.es/dspace/bitstream/10045/19601/1/1_http_y_aplics_web.pdf
14. Windows Server 2012 r2 → http://www.microsoft.com/es-es/server-cloud/products/windows-server-2012-r2/
15. Microsoft office Project Standart 2013 → http://www.microsoftstore.com/store/mseea/es_ES/pdp/Project-Standard-2013/productID.263156800&gclid=CLit0_jE-8UCFSUDtAodZ7YAhw&gclsrc=ds&tduid=92063ff2bff278f86480112209743062
16. W3school → http://www.w3schools.com/
17. PHP → http://php.net/
18. Ruby On rails → http://rubyonrails.org
19. MySQL → http://www.mysql.com/